Towards a semantic network of event's data from the web

Ayala Gómez, F.

Award date:
2014

Eötvös Loránd University
Faculty of Informatics

Towards a Semantic Network of Event's Data from the Web

Dr. András Benczur
Assistant Professor at ELTE
Dr. Zoltán Istenes
Assistant Professor at ELTE
Dr. Mykola Pechenizkiy
Assistant Professor at TU/e

Frederick Ayala Gómez
Computer Science

Budapest, 2014.

Abstract

Internet is an interesting marketing opportunity for events. However, users are required to visit several web sites to navigate the large amount of content that these organizations are promoting. In this thesis we present how information and communication technologies can help users to navigate better through this content.

Extracting and structuring social events from the Internet is the first step to support the users to navigate in event information. Current research on event detection has succeeded in detecting social events by using general text streams, social media and multimedia [SWC13, ACD$^+$98, APL98, YY98, KA04, ZZW07, WL11, ZC13, RPP$^+$13]. The contribution of this thesis is a software architecture that considers different characteristics by applying data mining and information retrieval techniques to create a graph of social events.

The presented work is organized as follows. First we introduce the topic of event detection, the and the particular research objectives of this thesis. The objectives are reached by designing a software architecture that we use to validate the hypotheses by experimentation. Finally the results of the experiments are analyzed to draw conclusions.

The domain of this work is limited to organizations that promote social events over the Internet with a specific date, time and location. Particularly, the focus is on concert halls, theaters, museums, festivals, nightlife and free-time activity providers.

The objectives of this research are the following. Identify the official web sites of the event organizers. Detect the HTML documents in the web site that contains events. Extract the information of the events. Improve the data quality and finally, create a graph of social events. The software architecture and hypotheses are aiming to satisfy these objectives.

The experiment applies the ideas proposed in the software architecture for the city of Budapest, the capital of Hungary. The social events detected were compared with the official profiles from Facebook, Last.fm and Eventful of the event organizers.

We conclude that the software architecture and the ideas behind it are valid and useful for structuring social events from the Web. Further work can include an extensive benchmark of different methods for event detection and more extensive experiment in different contexts.

The execution of the software tools required in this project run in the infrastructure provided by MTA Sztaki in the Data Mining and Search Group.

Dedicated to my parents Marina and Gaston for their support, love and care during my whole life. To my sister Monica and my brother Alfonso for teaching me to carry on and stay focus on my goals. To my friends, particularly to Andres, Ramón, Diego, Daniel, Raúl and Alfredo for their motivation to study my Master Degree in Europe.

Acknowledgments

First of all, I wish to express my honest gratitude to the EIT ICT Labs master school for organizing this program. Having the opportunity to study a master degree in Europe was a very exciting period in my life and the lessons learned are unforgettable. I do believe that connecting students, universities, research institutes, private companies and different cultures can lead to innovations that deliver positive impacts to society.

I also want to express my sincere thanks to the lecturers, professors and administrative staff of both universities. Eindhoven University of Technology (TU/e) in The Netherlands and Eötvös Loránd University (ELTE) in Hungary.

In relation to my studying period in Eindhoven. I want to express my thanks to Assistant Professor Dr. Bas Luttik for his advices during my studies at TU/e. Thanks to Assistant Professor Dr. Mykola Pechenizkiy for co-mentoring my thesis and inspired me to study further the fields of information retrieval, data mining and web engineering. Also, to the EIT Co-location Center in Eindhoven. To the Node Director Patrick Strating, CLC manager Erwin van Eijden, Manager Marketing Communications Else Embregts, Action Line Leader Health & Wellbeing Jean Gelissen and Office Manager Dorine Hamers. Thanks to all of you for your help and support.

From ELTE University I want to express my thanks to the Faculty Dean Dr. Zoltán Horváth, Associate Professor and Head of Department Dr. Zoltán Istenes, and Associate Professor and Vice-Dean Dr. Tamás Kozsik for their advice during my studies at ELTE. Also, to the EIT ICT Labs Budapest Associate Partner Group. To the Deputy Director Zsuzsa Várhalmi, Student Advisor Zsófia Lovas and Marketing and Communications Manager Bálint Molnár. Together, you provided me with great experiences, learning and support.

I also thank to the Head of Department at the MTA SZTAKI Informatics Laboratory Dr. András Benczúr, for mentoring me during my internship at the Data Mining and Search Group. In general, thanks to everyone in this group for your kind help during my internship and thesis. Special thanks to Ph.D. student Róbert Pálovics and Ph.D. student Bálint Daróczy for providing me with knowledge, feedback and support.

Finally, I place my sense of gratitude to those who kindly helped me directly or indirectly during my studies, internship, thesis and life.

Acronyms and Abbreviations

CART	Classification and Regression Trees
CRFs	Conditional Random Fields
CSS	Cascading Style Sheets
DARPA	Defense Advanced Research Projects Agency
DM	Data Mining
DOM	Document Object Model
ER	Entity Resolution
FSM	Finite State Machines
HITS	Hiperlink-Induced Topic Search
HMM	Hidden Markov Model
HTML	HyperText Markup Language
IR	Information Retrieval
ITU	International Telecommunications Union
JMX	Java Management Extensions
ML	Machine Learning
NED	New Event Detection
NER	Named Entities Recognition
NLP	Natural Language Processing
SVM	Support Vector Machines
TDT	Topic Detection and Tracking
WARC	Web ARChive
XML	Extensible Markup Language

Contents

List of Figures

List of Tables

Introduction

Internet is turning to one of the biggest media in the world and everyday is competing more with television, radio and printed media. With the amount of users that the Internet has, organizations are interested in using this media as a marketing channel.

Event organizers use a wide variety of Internet sources to promote events. While this may help them to to reach a bigger audience, it creates two problems. The first is that the sources of information may lose synchronization. The second is that the information gets distributed among different web sites.

Because of this, users need to invest time and effort to navigate among several web sites to understand the social events. However, information and communication technologies can play a key role to help users to understand better this information. This is the motivation to focus on those organizations that promote social events over the Internet.

This thesis presents an approach to create a graph of the social events from the Web. We can imagine this graph as a structure where different organizations (e.g. museums, concert halls, galleries, night clubs) are connected to the events they are promoting on Internet. Additionally, connections between events that share the same characteristics are created in this graph (e.g. genre, artist, location).

The methodology of this work is the following. First we define the objectives of the research. Then, by designing a software architecture we aim to to satisfy the objectives. Based on the software architecture an experiment is done. Finally, we draw the conclusions of the objectives, software architecture and thesis work in general.

The document is organized as follows.

Chapter 1 is an introduction to the Internet as an important media to advertise events. The concept of event detection is presented together with the current research in this field. Finally, the objectives for this thesis are presented.

Chapter 2 provides the theoretical background that used in the even detection system. Including the definition of graphs, graph link analysis, data mining classifiers, Named Entities Recognition and Entity Resolution.

Chapter 3 presents the hypotheses and a system that integrates different algorithms to accomplish the objectives of this thesis work.

Chapter 4 shows an experiment designed to test the ideas of the proposed system. The results of the experiment are presented and compared with Facebook, Last.fm and Eventful.

Chapter 5 concludes the thesis work by analyzing the results and verifying that the objectives were achieved. Finally, the possible future work, weaknesses and limitations are presented.

Chapter 1

Social Events Detection

The first part of this chapter defines the concept of event detection and gives an introduction to the Internet as a media source for event advertising. After, we review the current research on event detection that uses general text streams, social media feeds and multimedia content. The last part of this chapter presents the thesis objectives: Trusted information sources detection, event identification, event information extraction, quality improvement and social events graph construction.

1.1 Using Internet for Event Advertising

Let us start with the definition of an event.

Concept 1. *An event is a social gathering that involves two type of participants: hosts and guests. Hosts are organizations that offer an activity or activities to guests with a given scope, date, time and location.*

According to the ITU, around 39% of the world population had Internet access in 2013 [Uni14]. Additionally, since 2002 Internet technologies such as Web 2.0, Social Networks and Audio/Video Streams enabled more interaction between the users and the providers to create much richer content [TO09].

The book "Event Marketing: How To Successfully Promote Events, Festivals, Conventions, And Expositions" [Hoy09] explains that the Internet is turning to one of the biggest media for event advertisement, competing with television, radio and print media. Table 1.1 shows the advantages that Internet brings to event organizers.

Advantage	**Description**
Brand building	Establishes an instantly recognizable brand in the website.
Direct marketing	Features for personalized marketing are available without additional costs.
On-line sales	Internet websites can enable payments for tickets, reservations, etc.
Customer support	Easy access to frequently asked questions.
Marketing research	The users preferences, demographics, web site usage and other information can be used to gain feedback from the market.
Content publishing services	Web sites can be used for publishing, editing and modifying content for the Internet users.

Table 1.1: Advantages of the Internet for event organizers.

Table 1.2 displays some fictional organizations that are of our interest and that use Internet to promote their events.

Organization	**Description**
GetLocal Tourist Information	A company that promotes events and activities organizers.
Alice Concert Hall	A company that promotes concerts.
Bob Museum and Alice Gallery	Promote different kind of artistic events.
AdventureWorks Cinema	A place that offers movies to the audience.
Jazz Night Lounge Club	A night club that offers events from time to time.
Sport Pub	A pub that promotes particular events.
Awesome Festival	Annual festival of awesome artists and shows.

Table 1.2: Examples of organizations.

Each of these organizations has a different size and a different number of employees, marketing budget and audience. Therefore, they use different kinds of Internet sources to promote their events. Figure 1.1 provides an example of some Internet sources that a company might use.

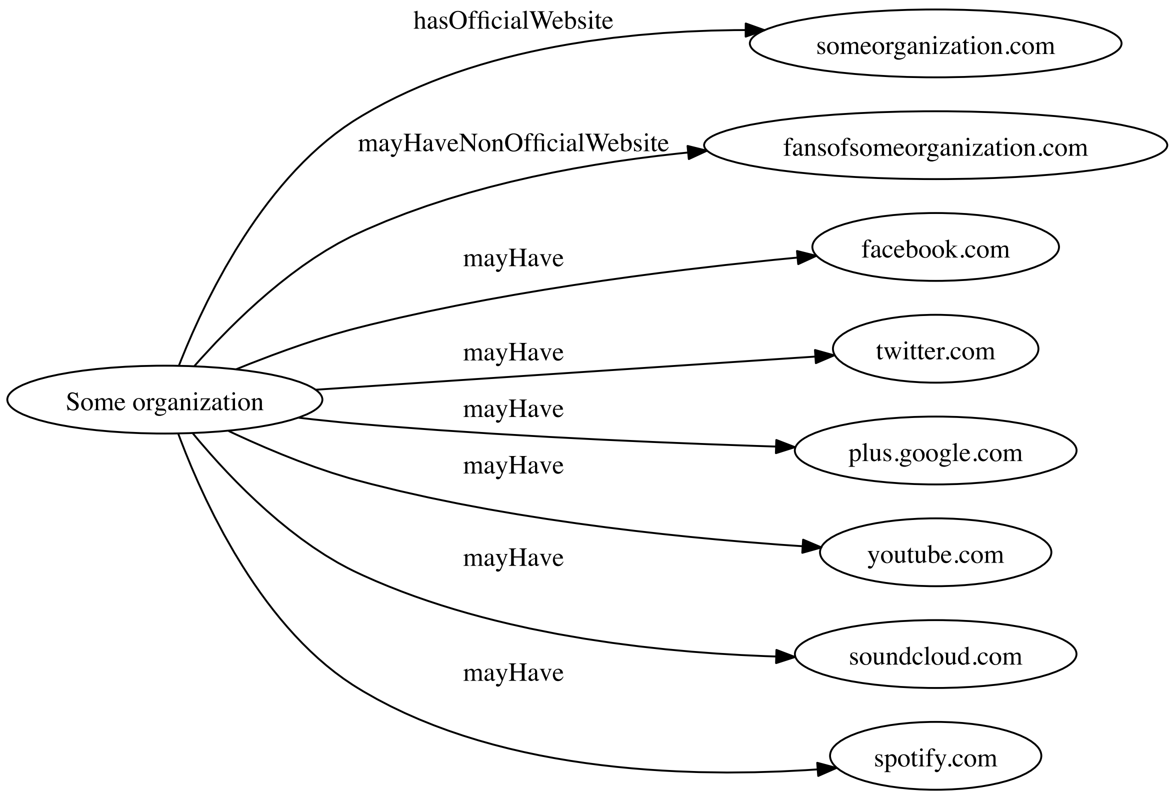


Figure 1.1: A company may have different Internet sources to promote events.

Having a wide variety of Internet sources to promote events may help an organization to reach a bigger audience. However, publishing content in several sources can have two problems. The first is that the sources lose synchronization. That is, one source may contain more up-to-date information than the others. The second is that the information gets distributed among different sources. This affects mainly to the end user since navigating among all the different Internet sources takes time.

The information presented in these Internet sources may be extensive. Usually, a social event involves multiple artists, genres, location, ticket prices, accommodation, transport and so on. If we combine the complexity characteristics of social events together with the distribution of information among different websites, it turns out that the ability of Internet users navigate social events is affected.

To reduce the complexity of event information access on the Web, Information and communication technologies can be applied. Particularly, the fields of Natural Language Processing (NLP), Information Retrieval (IR) and Data Mining (DM) can be used to retrieve, process and recommend content about social events to the user and enable them to navigate the content in a much simpler way.

1.2 Current Research on Event Detection

In this section, the current research on the domain of event detection is presented. These approaches focus on general text streams (e.g. news, articles), social media streams (e.g. Twitter) and multimedia (e.g. Flickr) to detect events.

First, it is important to mention two points. The first is that the work in this thesis is aiming to contribute, to the current research by proposing a software architecture. Instead of focusing on the detailed design and implementation, we focus on describing the objectives of the system and the activities that lead to satisfy our objectives. Thus, the implementation details of each activity are not part of the software architecture. Secondly, since a lot of progress have been made in detecting events, the software architecture makes use of many research results in the field of NLP, IR and DM.

1.2.1 General Text Streams

Researchers aim to detect events by using algorithms from NLP, IR and DM in the text of web documents, news and magazines. There are different approaches to solve this task. In this section we review some of them together with the objectives of the task.

Let us start by reviewing an initiative organized by DARPA with participation from Carnegie Melon University, Dragon Systems and the University of Massachusetts at Amherst [ACD$^+$98]. The objective of this initiative was to identify the state of the art on Topic Detection and Tracking (TDT). The data set for the research consisted on CNN and Reuters news. The research was focused on particular tasks and each of the participants proposed a different approach to solve it. These tasks are described in Table 1.3.

Task	**Description**
Segmentation	Segmenting a continuous stream of text into its constituent stories.
Retrospective event detection	Retrospectively process a corpus of stories to identify the events discussed therein.
On-line new event detection	Identify new events as they occur, based on an on-line stream of stories.
Tracking	The tracking task is defined to be the task of associating incoming stories with events known to the system.

Table 1.3: Tasks in the DARPA initiative for TDT.

The conclusions of the paper about each task are collected in Table 1.4

Task	**Description**
Segmentation	Segmentation is a tractable task using known technologies (Hidden Markov Model (HMM), IR, Machine Learning(ML).
Retrospective event detection	Pure retrospective detection can be performed quite reliably for most events by clustering methods.
On-line new event detection	Online detection cannot yet be performed reliably.
Tracking	Several IR techniques can be applied to solve this task.

Table 1.4: Conclusions for general text streams.

Now let us introduce the research presented in [APL98]. Related to the TDT initiative and focusing on on-line new event detection and tracking. By using a single pass clustering approach, they showed that event detection has a reasonable solution. The paper [YY98] focuses on retrospective event detection. They investigated two methods. The first is an agglomerative hierarchical clustering based on group-average clustering and the second is a single pass incremental clustering algorithm . The paper concludes that both methods work well for retrospective event detection. However, for on-line new event detection, non-clustering algorithms perform better.

The paper [KA04] discuss the usage of classification algorithms together with named entities to improve event detection. Instead of using the cosine similarity used in [APL98, YY98], they decided to base the similarity on another parameters. These can be category (e.g. finance, accidents), the overlap of named entities and non-named entities and other parameters that help to decide whenever a story is new or old. The model was trained using BoosTexter [SS00]. They created a series of simple rules that are used to build the classifier using term frequency as features. Named entities were identified by using the *BBN Systems & Technologies* Identifier [BSW99]. The entities of their interest were event, language, location, nationality, organization, person, cardinal, ordinal, date and time. The conclusion of this paper is that classification and named entity recognition may be directions for New Event Detection (NED) technology. Additionally, stop lists and temporal information should be considered in NED systems.

The paper [ZZW07] introduces three approaches to improve NED. The first is to use a news indexing-tree created dynamically to improve the running time by reducing the number of comparisons. To do this, a tree is created by analyzing how similar the news are to others in different hierarchies in the tree. The second is to use clustering for building the first level of the tree that represents the news topic. Finally, they detect the type of the different terms (e.g. noun and verb). These types are used to detect how they are related to different classes of stories to determine whether two stories are on the same topic. They conclude that their model improves both accuracy and execution cost of any of the others approaches.

Finally, in the paper [SWC13] they compare two NLP techniques: Finite State Machines (FSM) and Hidden Markov Model (HMM). The objective is to identify which of them performs better for the extraction of event related contextual information. Their experiments show that both methods produce high accuracy. However HMM performed well for *event title* extraction and FSM proved to be better for *venue*, *date* and *time*.

1.2.2 Using Social Media Feeds

Besides general text streams, researches make use of popular social networks such as Twitter to identify events. One of the greatest benefits of social networks and particularly in microblogs is the speed of published content. Also, meta-data such as location or hashtags are relevant. However, there is an extensive amount of information and many of the posts are useless and very short (e.g. under 140 characters on Twitter).

In the paper [WL11] researchers used Twitter data as an important source of events. The approach presented for event detection used wavelet-based signals. This algorithm builds signals based on the individual frequency of the words. Wavelet analysis provides precise measurements regarding when and how the frequency of the signal changes over time. Then, the trivial words are filtered. To identify the event they cluster the signals. Finally the number of words and cross correlation measure the relevance of an event. They conclude that this technique can be successfully used to identify events and considering additional features can do further improvement.

In [ZC13] researchers propose a graphical model–based framework. First they propose a location-time constrained topic graphical model to capture the social data information over content, time, and location. Second, they measure similarity between two messages by using Kullback–Leibler divergence-based measure. Third, they compute a link analysis measure using a longest common subsequence. Finally, they aggregate these two measures by extending them to weights for the message similarity. The model was tested in two large social media data sets resulting in a proven efficient and effective approach to detect events.

1.2.3 Using Multimedia Content

Another way to detect events is using social multimedia providers such as Flickr. Flickr is an image and video hosting website that enables for users to share and embed personal photographs in an on-line community.

The MediaEval Benchmarking Initiative for Multimedia Evaluation is an organization that aims to evaluate new algorithms for multimedia access and retrieval. They organized different challenges for social event detection [PSM$^+$12, RPP$^+$13] resulting in different approaches to successfully address this task. Each of the challenges include a data set that contains photos obtained from Flickr labeled to their social event.

Starting with [PPK12], researches used multimodal clustering to solve the problem of social event detection. The dataset used was the data from MediaEval social event detection challenge [RPP$^+$13]. The paper uses a multimodal clustering. First, the algorithm computes the pairwise similarities for all the modalities of two items. Similarity is used to determine if they belong to the the same cluster. Then, a classifier is trained using the labels in the dataset to predict the same cluster relationship. Finally, all the pairwise of the same cluster are used to obtain the final multimodal clustering. They conclude that the multimodal clustering approach can be applied to the problem of social event detection.

The paper [RC12] presents a system that is able to classify a stream of social media data into a growing and evolving set of events. The dataset used in this paper was the data from MediaEval social event detection challenge [RPP$^+$13]. Their paper proposed two steps. The first is to include a candidate retrieval step to select candidates that are related to the new incoming data. The second step is to determine if the new event refers to a new event or to a top the scoring retrieved candidate. This paper concludes that the techniques they used led to a state-of-the-art approach in terms of quality and scalability since they were able to have a high score in candidate retrieval (99.9%), scoring and ranking (99.4%) and new event detection (85.9%) while keeping high precision.

1.3 Objectives

The opportunities and benefits that Internet brings to organizations for events promotion were explained in the first section of this chapter. Also, we reviewed the current research on event detection. In this section we present the research objectives of this thesis.

1.3.1 Organizations Detection

Social events are limited to geography, location and time. Thus, we need to specify the kind of organizations we are interested in. We focus on organization in at least one of the categories described in Table 1.5.

Category	Description
Concert Halls	A large public building designed for the performance of concerts.
Theaters	A building or outdoors area where plays and other dramatic performances are given.
Museums and Galleries	A building where objects of historical, scientific, artistic, or cultural interest are stored and exhibited.
Festivals	An organized series of concerts, plays, or movies, typically one held annually in the same place.
Nightlife	Mainly focused on clubs, restaurants and bars that offers entertainment mainly for adults.
Free-Time	Organizations that offer activities for spare time (e.g. tourist walk guides, cycling trips).

Table 1.5: Organizations categories.

Objective 1. *Identify the event organizers that fit in at least one category of our interest and that use Internet to promote events.*

1.3.2 Trust

In general, the perception of trust strongly influences the user behavior on Internet [SH02]. However, trust might be a vague concept since it can be seen from different points of view. To avoid this, in the paper written by B. Suh and I. Han [SH02] they present a summary of the four trust classes proposed by McKnight and Chervant [MC02]. Table 1.6 shows these classes.

Trust Class	Description
Disposition to trust	Is the level that one is willing to depend on others in general across a broad spectrum of situations and persons.
Institution based trust	One believes that the conditions that lead to success are in place.
Trusting belief	One believes that the other party has one or more characteristics beneficial to oneself.
Trusting intention	One is willing to depend on, or indents to depend on, the other party even though one cannot control that part.

Table 1.6: Classes of trust.

Since the idea of creating a semantic network of events brings benefits to the parties involved, trust in this thesis is placed in the category of "Trusting Belief".

Concept 2. *Structuring the event's data into a semantic network should create one or more characteristics that are beneficial for the interest of their users.*

It is known that there are Spam and Phishing techniques trying to manipulate how users and web crawlers process the data from the web. Therefore, mechanisms to manage trust are required. The authors Ninghui Li and John C. Mitchell discuss a framework [LM02] to manage trust via role-base. There are two ways of managing trust: Reputation Trust Management and Rule Trust Management.

Reputation Trust Management

In [ZM00] two definitions for reputation are presented. Reputation is usually defined as the amount of trust inspired by a particular person in a specific setting or domain of interest [Mar00]. Reputation is regarded as asset creation and it is evaluated according to its expected economic returns [Rea96].

Additionally, reputation is a measure that needs to be defined and understandable among the parties involved. Moreover, managing trust by reputation is available in open systems (i.e. different domains). For each domain, an individual may have different reputation ranking. In reputation trust management, trust is a measure that changes in time. Because of this, reputation should be tracked and updated during the lifetime of the system.

Rule Trust Management

A rule means that the subject to be trusted holds a credential. A credential is a statement signed by an issuer, about a subject, and contains information about this subject. Credentials must be unforgeable, verifiable, signed and with well-defined semantics. The one who is accepting the credential should explicitly specify the issuers allowed to accredit subjects.

Objective 2. *Based on reputable web sites (e.g. official tourist websites) we detect the web sites of organizations that promote events.*

1.3.3 Event Information Retrieval

One of the fundamental objectives of this thesis is to detect that part of a web document represents an event.

For a user, it is easy to identify if a website contains data about events. For example, a user can go to the events or calendar subpage of a website and understand that the text there is describing an event. For a computer it is not that simple. First of all, HTML documents contain unstructured data. Even though HTML is a mark-up language, this does not mean that there is a definition of what kind of data the document contains. Moreover, although it is possible to extract the text from an HTML file, it is non- trivial to understand what the text is talking about.

For example, consider the text *Friday 12th of April, Rush, 20 00*. A computer can understand that Friday 12th of April is a date and that 20:00 is time. At this point we know that probably something will happen on that day and time. However, *Rush* can have different meanings (e.g. movie, artist). Because of this, we divided the information retrieval objective in two: Event detection and event information extraction.

Events Identification

Objective 3. *Identification of those parts of a HTML document that might describe an event.*

Event Information Extraction

This objective refers to the event information that the process needs to identify. For all the activities, we need to detect the location, name, data and time of the event. However, the price of the ticket it is not mandatory since it is not always present. Also, the information about the events vary according to event category. Table 1.7 contains optional information that we are interested to retrieve.

Organization	Optional Data
Concert Halls	Lineup of the concert. If the concert is part of a Tour.
Theaters	Name of the play, actors and description.
Museums and Galleries	Exhibitions name, duration and artists.
Festivals	Lineup.
Night-Life	Event description.
Free-Time	Event description.

Table 1.7: Optional named entities.

Objective 4. *Extract the required and optional event data and name the entities accordingly (e.g. artist, location, ticket price).*

Figure 1.2 is an example for this objective and presents and what the solution may look like.

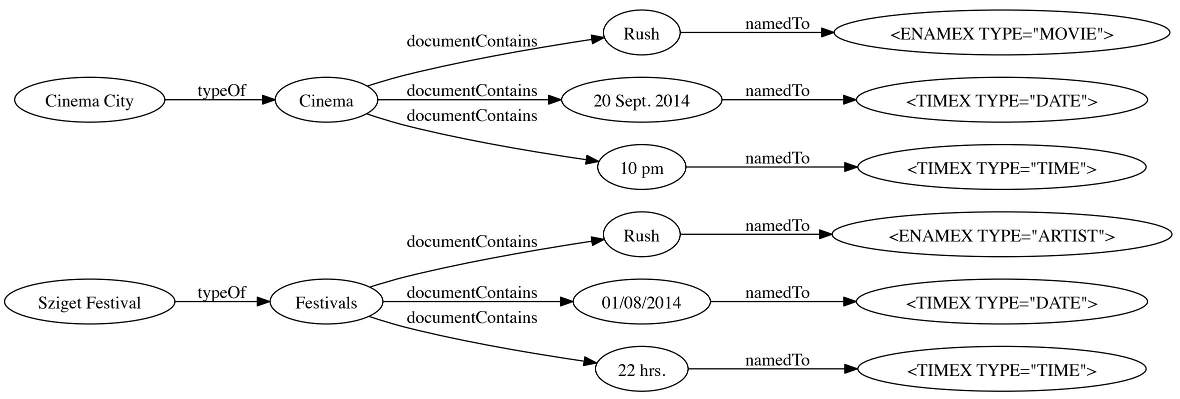


Figure 1.2: Example of event information retrieval.

1.3.4 Data Quality Improvement

Let us assume that the event information retrieval objective is solved and we have the information stored somewhere. Since there are many ways to describe the same information (e.g. dates, addresses), entity resolution aims to combine all the information that is the same into the same concept or category [SD06]. The aim is to enhance quality and improve the structure by avoiding duplicates. Figure 1.3 is an example of how the information can be combined to improve the quality of data.

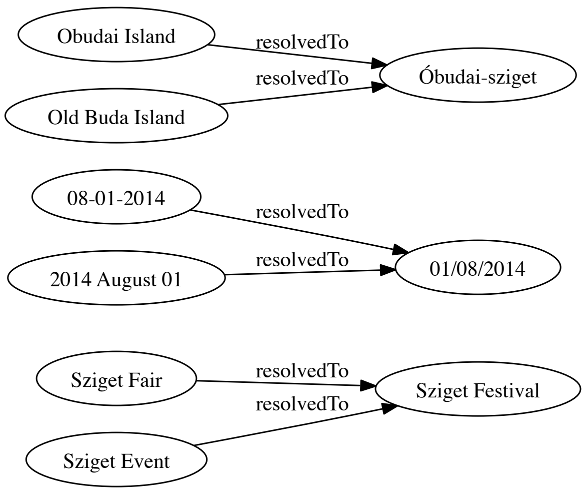


Figure 1.3: Entity Resolution.

Objective 5. *Enhance the data quality of event detection to improve the outcomes of the process.*

1.3.5 Events Graph Construction

The very last objective of this thesis is to build a semantic network. In this thesis we use semantic network, graph and network as the same concept. The concept of semantic network can be found in the *Encyclopedia of Artificial Intelligence* [Sow91]:

"A semantic network or net is a graph structure for representing knowledge in patterns of interconnected nodes and arcs."

Computer implementations of semantic networks were first developed for artificial intelligence and machine translation. But earlier versions have long been used in philosophy, psychology, and linguistics [Sow91]. Our aim is to a networks that is able to express the knowledge about an event by using the information from the previous objectives.

Figure 1.4 is an example of a graph expressing the relations of locations, events and their types.

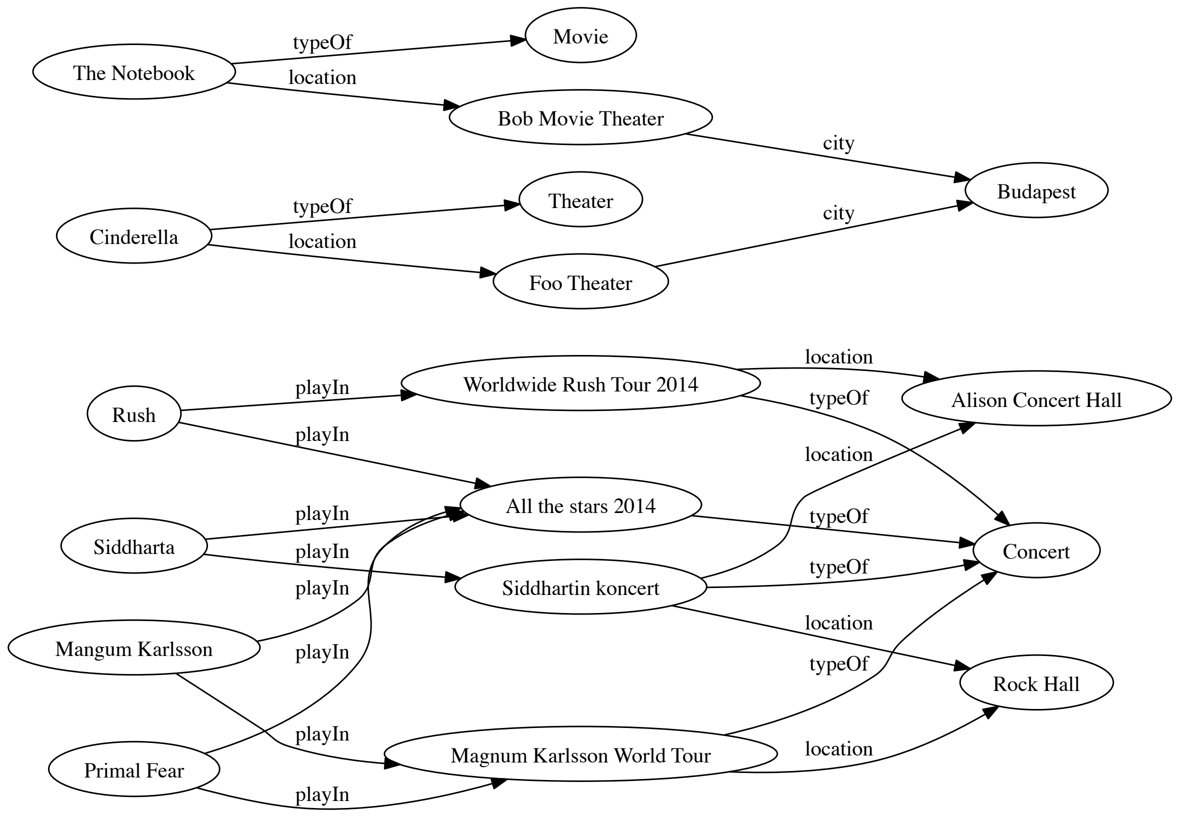


Figure 1.4: Social Events Network Example.

Objective 6. *Create a graph of social events using the resulting information of the previous objectives. Complement the resulting graph with additional information available on the Internet.*

In this chapter we introduced the concept of event detection and gave an introduction to the Internet as a media source f or event advertising. Also, we presented the current research on event detection and the objectives of this thesis. In the next chapter we review the background theory that allows us to structure social events from the Web.

Chapter 2

Theoretical Background

In the previous chapter, the current research on event detection was presented and the objectives of this thesis were described. This chapter is intended to provide the theory of the methods that are used in the events detection software architecture. We review the definition of graphs, link analysis, data mining classification algorithms, named entity recognition and entity resolution.

This chapter is based on the definitions and concepts presented in the books [NFSC13] and [TSK05]. These books provide detailed explanation on graph theory, data mining and their applications.

2.1 Graphs

2.1.1 Motivation for Using Graphs

A simple way to think about graphs is to imagine a group of objects that are connected in some way. Many problems can be represented as graphs (e.g. Social networks, retail networks, chemical components). Because of the different fields where graphs are used, there is a diverse and large amount of research on graph theory and applications from different areas (e.g. Mathematics, chemistry, physics, biology, geography, computer science, sociology).

In this thesis, we focus on a particular research area called *Graph Data Analytics* also known as *Graph Mining*. This concept refers to the extraction of insightful and actionable knowledge from graph data [NFSC13].

By detecting certain patterns in a graph, we identify relevant objects that can affect the network more than others. Examples for *Graph Mining* applications are described in Table 2.1 [NFSC13].

Area	**Application**
Web Graphs	The huge amount of information contained in the web can be expressed as a group of websites that are connected via hyperlinks. It is possible to look for sites that are referenced by others and those mainly reference other (i.e. authorities and hubs).
Social Science Graphs	Such graphs represent social relationships between people like friendship, family or neighbors. For example, it is possible to predict the probability of two individuals to become friends.
Finance Graphs	The structure of the stock market and trading records can be expressed as a graph. For example, nodes can be banks, brokers and investors. The connections between them can be the trading information (e.g. quote, amount, time). Thus, fraud detection can be performed to detect people involved in certain pattern of gain or loss.
Transport	A network of metro stations can be analyzed to identify stations that are more relevant to the transport system. By doing so, the organization can focus on keeping those stations in continuous operation.

Table 2.1: Examples of graphs applications.

Graphs represent structured data using vertices to describe information and the relation between them is symbolized by edges.

Definition 2.1.1. *Vertex*
A vertex (also called node) is a single point in a graph.

Definition 2.1.2. *Edge*
An edge in a graph G is an unordered pair of two vertices (v_1, v_2).

With these two definitions we can finally define a graphs.

Definition 2.1.3. *Graph*
A graph G is composed of two sets: a set of vertices, denoted $V(G)$, and a set of edges, denoted $E(G)$.

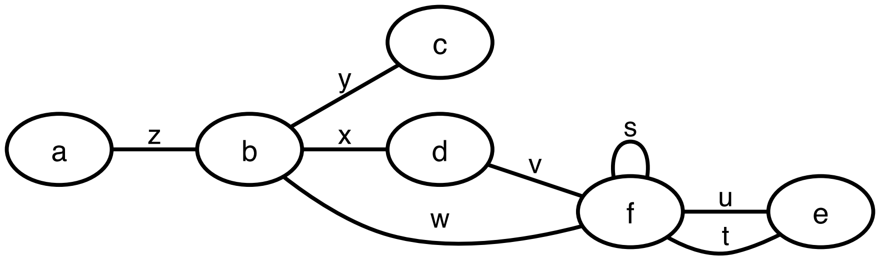


Figure 2.1: Example of a Graph.

An example of a graph is presented in Figure 2.1 where a, b, c, d, e, f are nodes and t, u, v, w, x, y, z are edges. In that graph it is possible to observe that the nodes e, f are connected by two edges t, u. This is called *Multiple Edge*. Also, we can introduce the concept of the *Degree*. The *Degree* of node b is 4 because 4 nodes uses it as an endpoint. The edge s from the node f to itself is called a loop and the degree of f is 5.

Definition 2.1.4. *Multiple Edge*
In a graph G, an edge is a multiple edge if there is another edge in $E(G)$ that joins the same pair of vertices.

Definition 2.1.5. *Degree*
In a graph G, the degree of a vertex v, denoted degree(v), is the number of times v occurs as an endpoint for the edges $E(G)$.

Definition 2.1.6. *Loop*
A loop is an edge that joins a vertex to itself.

Definition 2.1.7. *Simple Graph*
A simple graph is a graph with no loops or multiple edges.

2.1.2 Types of Graphs

There are many notions related to graphs and it is important for this thesis to know the following types: Subgraphs, isomorphic and automorphic graphs, directed graphs, cliques, paths and weighted graphs.

Subgraphs

Subgraph are graphs contained in another graph. In other words, we consider a subset of vertices and a subset of edges from a graph.

Figure 2.2 is an example of a subgraph from the graph in Figure 2.1.

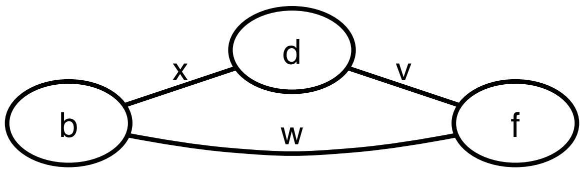


Figure 2.2: Graph example.

Definition 2.1.8. *Subgraph*
A subgraph S of a graph G is a set of vertices $V(S) \subset V(G)$ with a set of edges $E(S) \subset E(G)$. Every edge in $E(S)$ must be an unordered pair of vertices (v_1, v_2) such that $v_1 \in V(S)$ and $v_2 \in V(S)$.

Isomorphic and Automorphic Graphs

The concepts of isomorphic graphs and automorphic graphs are relevant when we want to compare graphs. Isomorphic is the case when the graphs have the same structure but different nodes. Automorphic is when the graphs have the same structure and same node labeling.

Definition 2.1.9. *Graph Isomorphism*
Two graphs G and H are isomorphic (denoted $G \simeq H$) if there exists a bijection $f : V(G) \rightarrow V(H)$ such that an edge $(v_1, v_2) \in E(G)$ if and only if $(f(v_1), f(v_2)) \in E(H)$.

Definition 2.1.10. *Graph Automorphism*
An automorphism of a graph G is an isomorphism f that maps G onto itself.

The concept of automorphism is exemplified in figure 2.3 with three graphs: A, B and C. The three of them are isomorphic but only graphs A and B are identical.

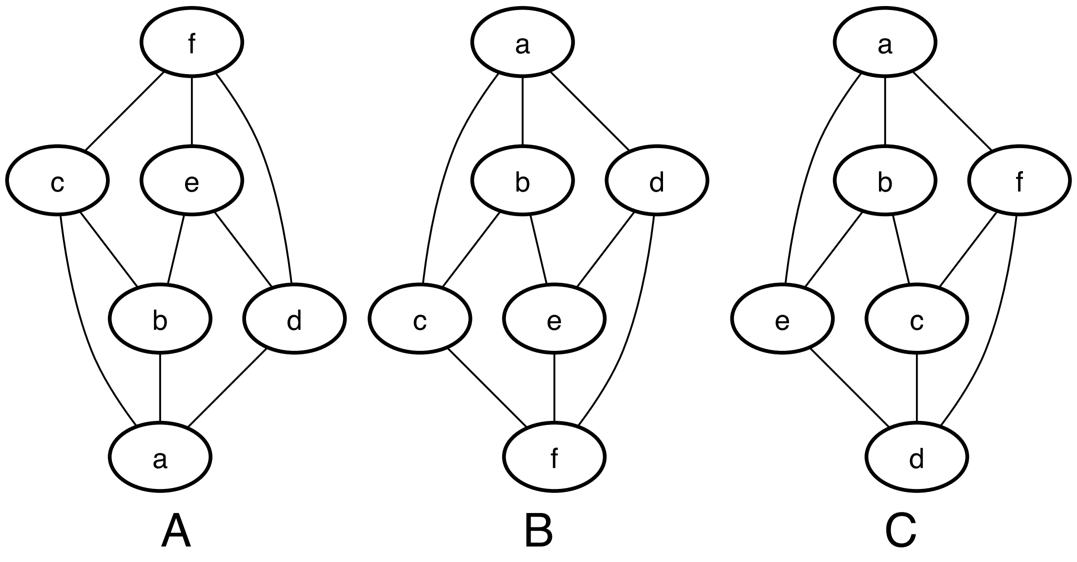


Figure 2.3: Three isomorphic graphs but only A and B are identical.

Directed Graphs

In the graphs we have reviewed so far the direction of the vertices does not matter. That is, we have consider that the direction between node (v_1, v_2) is the same that (v_2, v_1). In the case of directed graphs, the direction of the pair matters. The direction is usually expressed as an arrowed edge between the two nodes.

Definition 2.1.11. *Directed Graphs*
A directed graph D is composed of two sets: a set of vertices $V(D)$ and a set of edges $E(D)$ such that each edge is an ordered pair of vertices (t, h). The first vertex t is called the tail, and the second vertex h is called the head.

Figure 2.4 is a directed graph version of figure 2.1. In directed graphs you can move among nodes by following the edges from the tail to the head. It is not possible to move from the head to the tail of the edge.

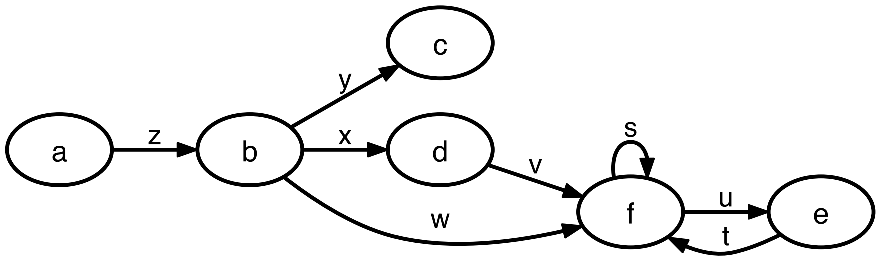


Figure 2.4: Directed graph example.

Cliques

A clique is a set of vertices that are all connected to each other by edges. In other words, a clique is a complete subgraph in G.

Definition 2.1.12. *Cliques*
A set of vertices C is a clique in the graph G if, for all pairs of vertices $v_1 \in C$ and $v_2 \in C$, there exists an edge $(v_1, v_2) \in E(G)$.

Figure 2.5 is a graph that contains a clique between the vertices a, b, c, d, e. Note that node f is only connected to vertices d, e therefore it is excluded from the clique.

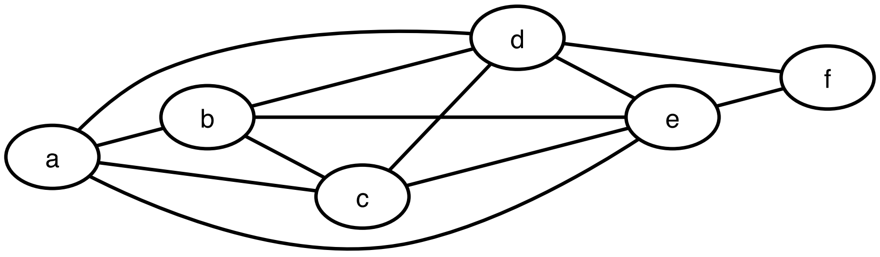


Figure 2.5: A graph containing a clique of 5 vertices.

Paths

Paths are used when we want to move from one node to another via the edges from the starting node to the ending node.

Definition 2.1.13. *Path(Vertices)*
A path of length n, denoted P_n, in a graph G is a set of unique $n+1$ vertices $v_1, v_2, ..., v_n, v_{n+1}$ such that for i from 1 to n, there exists an edge $(v_i, v_{i+1}) \in E(G)$.

Figure 2.6 is an example of a path from node a to f from the graph shown previously in figure 2.5. The length of this path is 3.

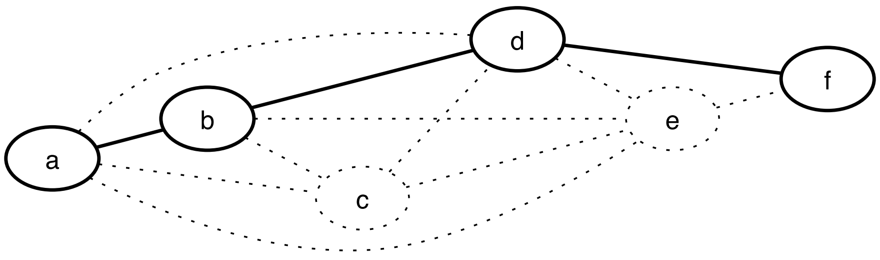


Figure 2.6: Example of a path from node a to f

Weighted Graphs

These kind of graphs are used to express the cost of moving from one node to another. The motivation for using weighted graphs is that we can define a metric to analyze the cost.

Definition 2.1.14. *Weighted Graphs*
A weighted graph W is composed of two sets: a set of vertices $V(W)$ and a set of edges $E(W)$ such that each edge is a pair of vertices v_1 and v_2 and a numeric weight w.

Figure 2.7 is an example of a weighted graph. We can observe that moving from a to f implies a lower cost via the path (a, b, e, f).

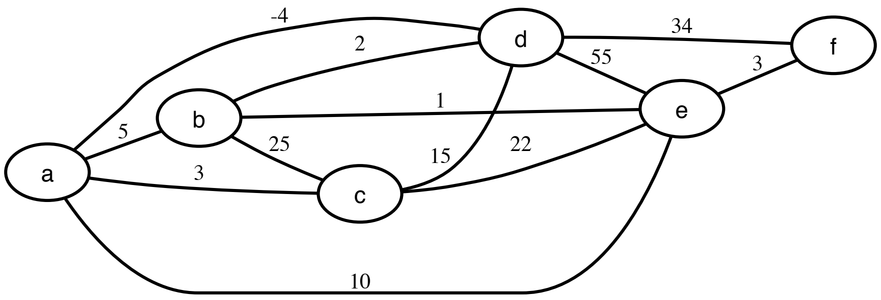


Figure 2.7: Example of a weighted graph.

2.1.3 Graph Representation

In this thesis two graph representations are used: Adjacency list and adjacency matrix.

Adjacency List

Definition 2.1.15. *Adjacency List*
Given a graph G such that $V(G) = v_1, v_2, ..., v_n$, the adjacency list representation of G is a list of length n such that the i^{th} element of the list is another list that contains one element for each vertex adjacent to v_i.

The representation of Figure 2.4 as a adjacency list is show in Table 2.2.

a	b
b	c, d, f
c	
d	f
e	f
f	e, f

Table 2.2: Example of a graph as an adjacency list.

Compared to the adjacency matrix representation, this list requires less memory.

Adjacency Matrix

Definition 2.1.16. *Adjacency Matrix*
Given a graph G such that $V(G) = v_1, v_2, ..., v_n$, the adjacency matrix representation of G is a $n \times n$ matrix. If $a_{r,c}$ is the value in the matrix at row r and column c then

$$a_{r,c} = \begin{cases} 1 & v_r \textit{ is adjacent to } v_c \\ 0 & \textit{otherwise} \end{cases}$$

Table 2.3 shows the adjacency matrix representation of the graph visualized in Figure 2.4.

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Table 2.3: Example of a graph as an adjacency matrix.

An advantage of this kind of graph representation is that it is simple to identify if two nodes are connected. It is only needed to check if there is a 1 in the matrix position of the nodes.

2.2 Link Analysis

In the examples shown previously, the graphs are not big and we can understand the relation between the nodes. However, in graphs with many nodes and edges it is difficult to understand how the graph looks like and how the nodes are connected. For example, a graph of 1500 nodes may have 2,250,000 edges represented in an adjacency matrix of 1500×1500 dimensions. We can use link analysis on graphs by considering the edges (i.e. links). This can help us to understand the structure and identify certain properties of the nodes based on how they are connected to other nodes in the graph. In this section we review Hiperlink-Induced Topic Search (HITS) and PageRank.

2.2.1 Hiperlink-Induced Topic Search

The objective of HITS is to rank a node as an authority and a hub. The algorithm was proposed by Kleinberg in [Kle99] and it has been applied to rank web search results and filter email Spam machines [DS04].

The HITS algorithm proposes to rank a node (i.e. website) according to two categories, authorities and hubs. A node is an authority if it has many pages with high hub scores that all link to it. On the other hand, a node is a hub if it points to many other nodes that have high authority scores.

The algorithm starts by selecting an initial set of nodes and links that are represented in an adjacency matrix form. Then we define two vectors. One authority vector $a = (a_1, a_2, \cdot, a_n)^T$ and one hub vector $h = (h_1, h_2, \cdot, h_n)^T$ where n is the number of vertices in the graph. Then, we initialize the vectors to 1

$$a^{(0)} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, h^{(0)} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

These vectors are used in the iterations of the algorithm. The next operations are performed to update the authority and hub values.

Definition 2.2.1. *Update Authority Scores*
The hub scores from the previous iteration is used to calculate the authority scores in the current iteration. An individual authority score for vertex $i \in V(G)$ is updated as follows:

$a_j^{(k)} \leftarrow \sum_{(i,j)\in E} h_i^{k-1}$,

where k represents the current iteration and $k-1$ is the previous iteration. Using the adjacency matrix A, the authority vector update operation can be defined as:

$a^{(k)} \leftarrow A^T \cdot h^{(k-1)}$

Definition 2.2.2. *Update Hub Scores*
The authority scores from the previous iteration is used to calculate the hub scores in the current iteration. An individual hub score for vertex $i \in V(G)$ is updated as follows:

$h_i^{(k)} \leftarrow \sum_{(i,j)\in E} a_j^k$

where k represents the current iteration. Using the adjacency matrix A, the hub vector update operation can be defined as:

$h^{(k)} \leftarrow A^T \cdot a^{(k)}$

For each iteration the hub and authority scores are normalized.

Definition 2.2.3. *Normalizing hub and authority scores*
The weights are normalized to ensure that

$\sum_{a\in a} a^2 = \sum_{h\in h} h^2 = 1$

We divide each component of the vectors a and h by the sum of squares of their respective components. Formally, $a = \frac{a}{\|a\|}$ and $h = \frac{h}{\|h\|}$ where $\|a\| = \sum_{a\in a} a^2$

In summary, the HITS algorithm performs the following steps:

1. Select an initial set of hyperlinks.
2. Initialize hubs and authority vectors to 1.
3. Iteratively update hubs and authority vectors.
4. Normalizing vector scores.
5. Output Vectors.

Kleinberg [Kle99] mentions that 20 iterations are enough to get authority and hubs scores in this algorithm. This is because the scores do not vary very much after 20 iterations so it may be possible to conclude that the algorithm has converged.

In Table 2.4 the calculations for the graph in Figure 2.4 are shown.

Node	Authority	Hubs
a	0.000	0.000
b	0.000	0.665
c	0.299	0.000
d	0.299	0.395
e	0.223	0.395
f	0.878	0.046

Table 2.4: Authority and hubs score example.

Node b is ranked higher as a hub since it has edges to nodes c, d and f while node f is ranked higher as an authority since nodes b, d and e point to it.

2.2.2 PageRank

The objective of PageRank is to assign a numeric value that ranks the node according to the count and quality of "incoming links" or "backlinks". PageRank represents the probability of the node to be visited on a particular random walk [BP98]. PageRank computes the eigenvector of a matrix using the *power method*.

The PageRank algorithm is described in Figure 2.8 [NFSC13].

```
Algorithm: PageRank calculation of a single graph
Input: $G$—Directed graph of $N$ web pages
$d$—Damping factor
Output: $PR[1 \dots N]$, where $PR[P_i]$ is the PageRank of page $P_i$
Let $PP[1 \dots N]$ denote a spare array of size $N$
Let $d$ denote the probability of reaching a particular node by a random
jump either from a vertex with no outlinks or with probability $(1-d)$
Let $N(P_u)^+$ denote the set of pages with at least one outlink
foreach $P_i$ in $N$ pages of $G$ do
    $PR[P_i] = \frac{1}{N}$
    $PP[i] = 0$
end
while $PR$ not converging do
    foreach $P_i$ in $N$ pages of $G$ do
        foreach $P_j$ in $N(P_i)^+$ do
            $PP[P_j] = PP[P_j] + \frac{PR[P_i]}{deg(P_i)^+}$
        end
    end
    foreach $P_i$ in $N$ pages of $G$ do
        $PR[P_i] = \frac{d}{N} + (1-d)(PP[P_i])$
        $PP[P_i] = 0$
    end
    Normalize $PR[P_i]$ so that $\sum_{P_i \in N} PR[P_i] = 1$
end
```

Figure 2.8: PageRank algorithm.

Table 2.5 presents the PageRank calculations for the graph in Figure 2.4.

Node	PageRank
b	0.058
f	0.054
c	0.048
d	0.048
a	0.031
e	0.026

Table 2.5: PageRank example.

Node f is ranked higher because it has many adjacent edges and e is pointing back to it. Node e is the second ranked because f is pointing it back as well. Node a is the last lower node since there is no node that points to it.

2.3 Classification

Classification algorithms are used to predict the class of an object. In this section we introduce the following algorithms: Logistic linear regression, classification and regression trees and support vector machines. Most of the concepts and theory presented are taken from the book [HTF09] that provides a deeper background for the techniques.

2.3.1 Logistic Linear Regression

Logistic linear regression is a linear method for classification. In this algorithm we consider a binary dependent variable. The model predicts a probability between $[0, 1]$ and we can define a the probability cut-offs to assign the 0 or 1 class for a prediction.

The following is the formula of the model:

$$P(y = 1|x_1, \ldots, x_k) = \frac{1}{1+\exp(-(\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots+\beta_k x_k))},$$

where $x_1, x_2, \ldots, x_k$ represent the independent variables for the model and $\beta_0, \beta_1, \ldots, \beta_k$ are the coefficients for each independent variable. These coefficients are used to adjust the values of the independent variables for the prediction.

2.3.2 Classification and Regression Trees

The aim of Classification and Regression Trees (CART) is to predict the outcome of an observation by following a decision tree. The model creates a tree that contains conditional splits that should be followed to determine the class. The prediction of the observation is based on a training set. To build the CART tree, the algorithm splits on the independent variables of the model. CART is of our interest since the model does not assume a linear relation and it is interpretable [HTF09].

Figure 2.9 shows an example of the outcome of a CART Tree algorithm. The objective of this model is to predict if an e-mail is relevant to the user. The data is the Enron corpus presented in [CGHO]. This data contains a label indicating if an e-mail is relevant or not. Since in this example we are working with text, some preprocessing is required. The e-mail text is changed to lower case, punctuation and stop words are removed, words are stemmed and we keep only the 3% of common terms.

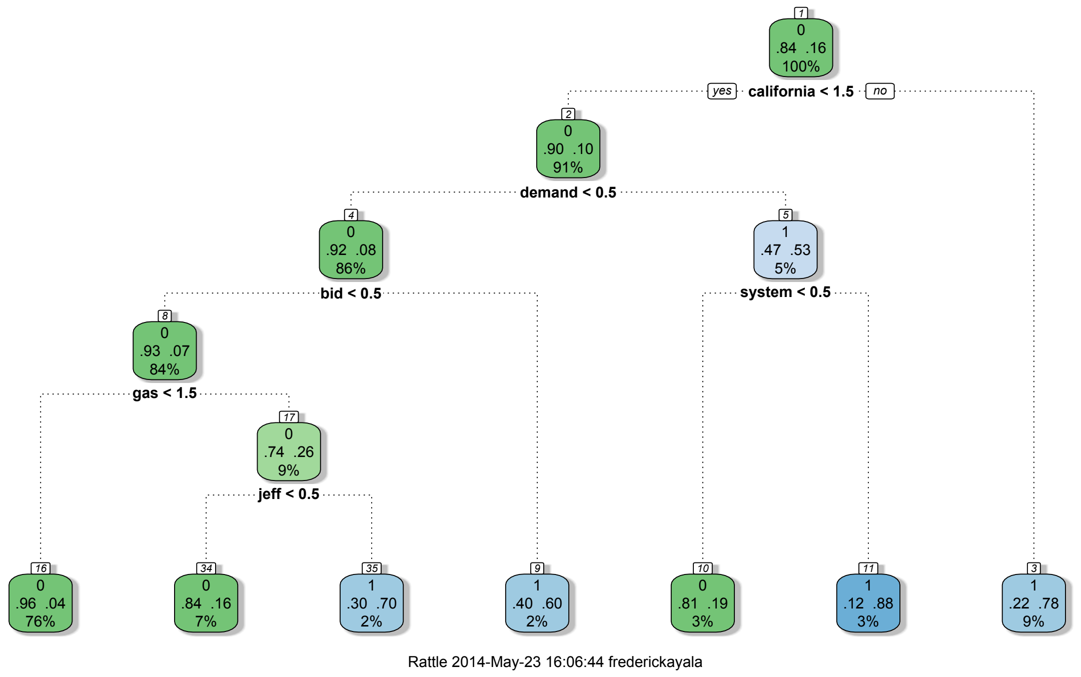


Figure 2.9: A CART tree example with the Enron dataset.

CART trees are read from top to bottom. The left direction means that the condition is true while the right directions means false. The ending node of the tree represents the most probable outcome for a given observation.

2.3.3 Support Vector Machines (SVM)

The last classification model presented in this chapter is SVM. This method has very good results in classification tasks like image and text classification.

SVM tries to find optimal hyperplanes for linearly separable patterns. Moreover, it can be extended to patterns that are not linearly separable by transforming the original data into a new space using Kernel functions. Additionally, it has very interesting properties that fit perfect for the classification problems presented in this thesis.

This method has a modular design that allows us to separately implement and design its components. For example, it is possible to choose different types of Kernels and cost functions. It also can be applied to categorical data and it does not suffer from the curse of dimensionality.

As presented in [TSK05], Figure 2.10 shows two classes, squares and circles. We can see that these classes are linear separable. However, finding the hyperplane that separates the lines best is not trivial.

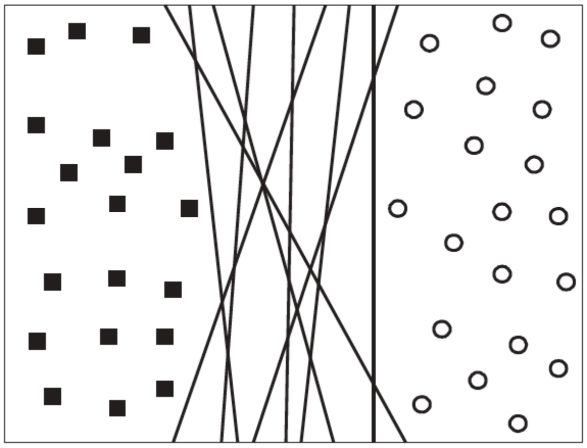


Figure 2.10: Different decision boundaries between the two classes.

The rationale behind finding a maximum margin hyperplane is that if the margin is small, there is not enough space to separate the classes. In other words, "Classifiers that produce decision boundaries with small margins are therefore more susceptible to model overfitting and tend to generalize poorly on previous unseen examples" [TSK05].

One of the linear classifiers that aims to identify a hyperplane with maximum margin is the linear SVM [CV95].

Given a set of n labeled examples $a_1, \ldots, a_n$ in d-dimensional space, the objective of this model is to find a d-dimensional vector w and a threshold b such that

$$w \cdot a_i > b \text{ for each } a_i \text{ labeled } +1,$$
$$w \cdot a_i < b \text{ for each } a_i \text{ labeled } -1.$$

A vector-threshold pair (w, b) satisfying the inequalities is called a linear separator. It is possible to rewrite this expression if we add an extra dimension to each sample and the norm.

$$(w' \cdot a'_i)l_i > 0,$$
$$\text{where } 1 \leq i \leq \text{n and } a'_i = (a_i, 1), w' = (w, b).$$

If the problem is linearly separable, $(w \cdot a)l_i > 0$ for all i.

For a solution w, where $|a_i| = 1$ for all examples, the margin is defined to be the minimum distance of the hyperplane $\{x|w \cdot x = 0\}$ to any a_i for a solution w.

Finally, let us consider two vectors a_2 and a_1 such that $a_2 - a_1$ is parallel to w and $wa_1 + b = 1$ and $wa_2 + b = -1$. We can solve the equation $a_2 - a_1 = -n\frac{w}{\|w\|}$.

The task of finding the hyperplane with a maximum margin is equivalent to minimizing the norm of the hyperplane vector,

$$\min_{w} \|w\|.$$

Figure 2.11 shows a diagram with the solution to the problem of finding the hyperplane with the largest margin between the support vectors a_1 and a_2.

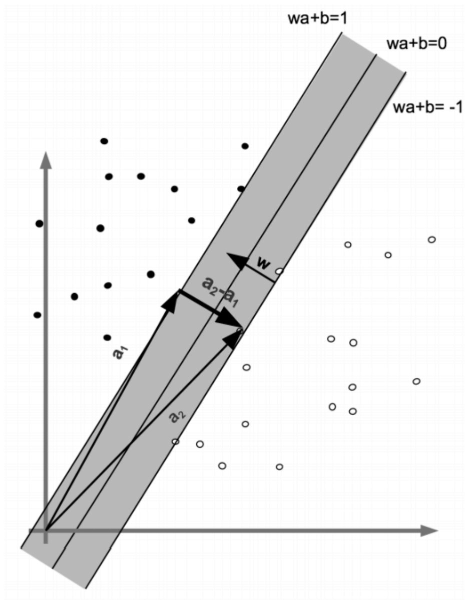


Figure 2.11: Support Vector Machine solution.

However, not all data can be separated linearly. In this case, SVM makes use of kernel methods to move the data into a higher dimensional space where it can be separated better.

Kernel methods increase the dimensional space of the original data allowing SVM to threat non-linear data as linear in a higher dimensional space [NFSC13]. By doing this, the *curse of dimensionality* is increased and the *curse of complexity* is reduced. The first one refers to the problem of performing statistical methods in high dimensions. The second one refers to the problem of performing complex algorithms. The *curse of complexity* tend to be more important than the *curse of dimensionality*.

Definition 2.3.1. *Kernel Method*
The transformation of data from the input space into a high-dimensional feature space, where analysis is performed.

Calculating the feature space in a high dimension can be computational intensive. To avoid this calculation, “it is possible to represent the algorithms using inner products. This strategy is known as the *kernel trick*. The function that computes the inner product in the feature space is known as a *kernel function*” [NFSC13].

Definition 2.3.2. *Kernel Function*
A symmetric, positive semi-definite function that represents inner product of the vectors in some feature space.

Definition 2.3.3. *Kernel Trick*
The use of kernel functions to allow linear analysis algorithms based on inner products to model nonlinear relationships in the input space.

Examples of kernel functions are: Polynomial, Gaussian RBF and Sigmoidal. The explanation of these is outside the scope of this thesis. In practice, most of the implementations of SVM support different kernels and one can pick the kernel that fits best the data. For example, the *R-package libsvm* [CL11] includes these kernels in the SVM implementation.

2.4 Named Entity Recognition

An additional task of this thesis is to identify the pieces of information in a social event. To accomplish this, the proposed event detection framework makes use of NER techniques.

At a glance, NER systems train a sequence model in the following steps. First, training documents are collected and parsed into tokents. Each token is marked with the right entity class or other (O). Then we design appropriate feature extractors. Finally, the training of a sequence classifier is done to predict the labels from the data.

Table 2.6 is an example of a manually labeled sequence of tokens. The encoding technique considers every token as independent and is called Input-Output encode (IO) encode. Of course, some objects (e.g. National Museum) are two-word. However, in practice it is better to use individual labeling.

Token	IO encoding
Scorpions	ARTIST
Concert	EVENTTYPE
in	O
Budapest	CITY
on	O
June	DATE

Table 2.6: Example of token labeling for NER.

The NER system that we use in this thesis is proposed in [FGM05]. The sequence training model it uses is called Conditional Random Fields (CRFs). This model has a better performance than other sequence models.

2.5 Entity Resolution

The last concept to review is Entity Resolution (ER). This process aims to identify groups of records that refer to the same real-world entity. This section is based mainly on [BGMM^{+}09].

Information systems might contain information that refers to the same real-life object but it is expressed in different ways. For example, information that has no standard, information that is incomplete, duplicates, typo mistakes or changes of the data with time.

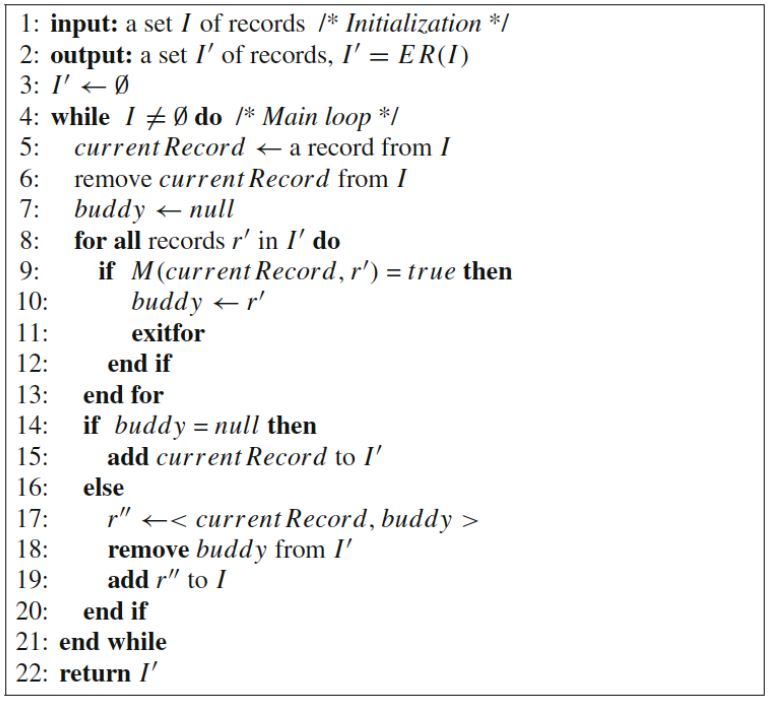
This thesis uses the work presented in [BGMM^{+}09] that is called *Swoosh*. The idea is to provide a generic approach for the problem of ER. To accomplish this, the researchers proposed the division of the problem in two. The first part are those functions able to match and merge records and the second is the ER algorithm that invokes these functions.

The functions able to match and merge records hold four properties shown in Table 2.7.

Property	Description
Idempotence	A record always matches itself. Merging with itself leads to itself.
Commutativity	The match of $(record_1, record_2)$ is the same than the match $(record_2, record_1)$.
Associativity	The match of $(record_1, (record_2, record_3)$ is the same that $(record_1$,$record_2), record_3)$.
Representativity	If $record_3 = (record_1, record_2)$ then for any $record_4$ such that $record_1 \approx record_4$, we also have $record_3 \approx record_4$.

Table 2.7: Properties for the *Swoosh* algorithms.

One of the *Swoosh* algorithms is called *R-Swoosh*. We use this algorithm to perform ER and the steps performed are described in Figure 2.12 [BGMM+09].



```
input: a set $I$ of records  /* Initialization */
output: a set $I'$ of records, $I' = ER(I)$
$I' \leftarrow \emptyset$
while $I \neq \emptyset$ do  /* Main loop */
  $currentRecord \leftarrow$ a record from $I$
  remove $currentRecord$ from $I$
  $buddy \leftarrow null$
  for all records $r'$ in $I'$ do
    if $M(currentRecord, r') = true$ then
      $buddy \leftarrow r'$
      exitfor
    end if
  end for
  if $buddy = null$ then
    add $currentRecord$ to $I'$
  else
    $r'' \leftarrow < currentRecord, buddy >$
    remove $buddy$ from $I'$
    add $r''$ to $I$
  end if
end while
return $I'$
```

Figure 2.12: The *R-Swoosh* Algorithm

Besides the *R-Swoosh* algorithm we use information of the graph to improve the ER result.

In this chapter we presented an introduction to graphs, graph link analysis, data mining classification algorithms, named entity recognition and entity resolution. These concepts can have many applications in different areas. In the next chapter we investigate how to combine these concepts to achieve our thesis objectives.

Chapter 3

Software Architecture for Structuring Social Events Data From the Web

The goal of this chapter is to use the concepts reviewed in the theoretical chapter in a system that is able to structure social event data from the web. To accomplish this, we use the 4+1 architectural view model [Kru95]. We review the logical view, process view, development view, physical view and scenarios of the proposed system. The main contribution of this thesis is to present a software architecture that describes a system from the viewpoint of one particular stakeholder. In our case we focus on a stakeholder called *expert* which is a person interested in structuring social events from the web and capable to provide judgment during the execution of the system.

3.1 Logical View

This view aims to show the functionality that the system provides to the end-user. In our case, the end-user is the person interested in structuring events from the Web. Figure 3.1 shows a class diagram of the system.

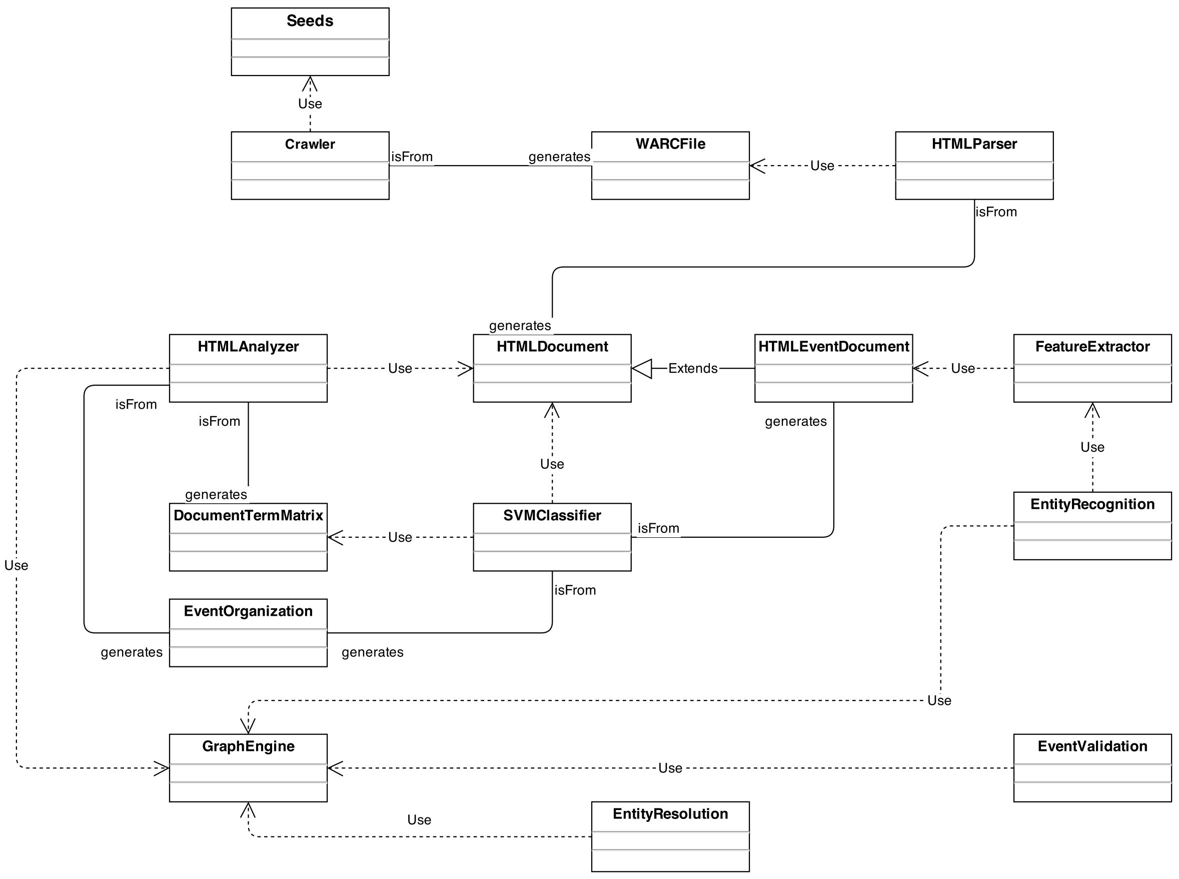


Figure 3.1: Class diagram.

We can see in the class diagram the relation between the classes that we use in the next views. The *Crawler* uses the *Seeds* to generate *WARCFiles*. An *HTMLParser* uses the *WARCFiles* to generate *HTMLDocuments*. These are used by the *HTMLAnalyzer* to produce the *DocumentTermMatrix* and the *EventOrganization*. The *SVMClassifier* uses the *DocumentTermMatrix* to generate the *HTMLEventDocument* and the *EventOrganization*. Then the *FeatureExtractor* uses the *HTMLEventDocument* to generate the features that the *EntityRecognition* uses. Finally, the *GraphEngine* is used by the *HTMLAnalyzer*, *EntityRecognition*, *EntityResolution* and *EventValidation*.

3.2 Process View

In this section, the processes of the system are presented. It is important to understand the activity sequence and the goal of each activity in the processes. Each process contains an activity diagram followed by a table describing the activities. Then, a hypothesis is formulated to connect the process with a research objective.

3.3 Organization Detection

The first task is to identify the organizations we are interested to detect events. To do so, the expert defines the seeds that are used to detect organizations. Then each of the seeds is crawled and processed to identify links to external websites. Finally, these links are classified according to the organization's type. Figure 3.2 presents the activity diagram of the process. The details of the activities can be found in Table 3.1.

ID	Activity	Description
1.1	Define seeds	An expert provides a list of domains that are considered the starting point for detecting external links.
1.2	Seed crawling	For each of the seeds, a crawl follows the internal links to retrieve the HTML documents of the website. Note that for efficiency this process can be done using parallel computing.
1.3	Generate WARC	The HTML documents are stored in a WARC file.
1.4	Process WARC File	An index of the WARC file is created and for each of the HTML documents the next activities are performed.
1.4.1	HTML parsing	The structure of the HTML document is parsed into its DOM representation.
1.4.2	Language detection	Since seeds can be in multiple languages, it is important to filter out those pages that are not in the language of our interest.
1.4.3	HTML analysis	The DOM Object is analyzed.

Table 3.1: Organization detection activities.

ID	Activity	Description
1.4.4	Links and text extraction	The external and internal links are extracted together with the text of each link and the full text of the HTML.
1.4.5	Graph insertion	Data is extracted and stored as a graph.
1.5	Seed analysis	Centrality measures, external links filtering and website detection.
1.6	Website crawling	The external websites are crawled at their first level.
1.7	Generate WARC	The HTML documents are stored in a WARC file.
1.8	Website labeling	An expert labels some websites according to the category they should be classified.
1.9	SVM model training	A SVM model is trained with the document-term frequency of the website and the labeled category of the expert.
1.10	Website classification	The class if the unclassified websites is predicted using the SVM model.
1.11	Graph update	The websites are inserted into the graph.

Table 3.2: Organization detection activities continuation.

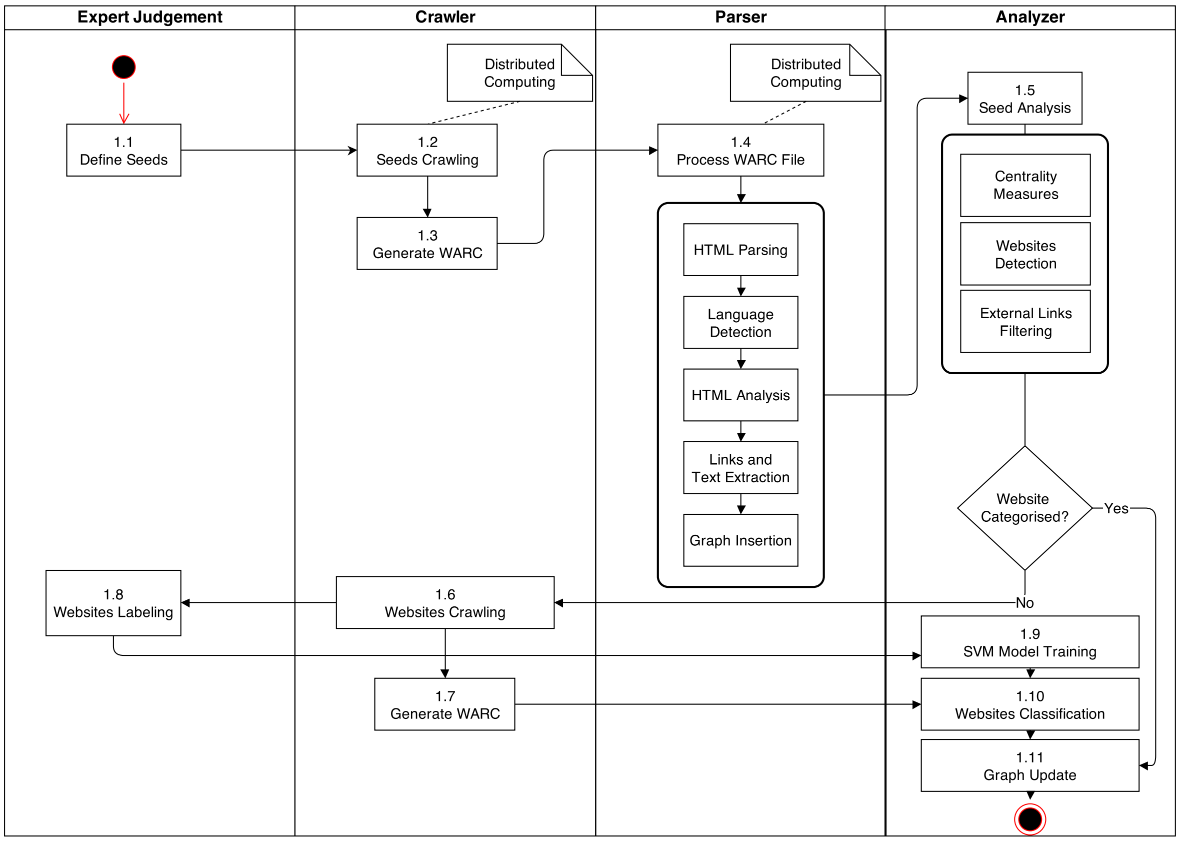


Figure 3.2: Organization detection activity diagram.

Hypothesis 1. *The process in Figure 3.2 allows us to crawl the Web based on reputable web sites. This identifies the event organizers that fit in at least one category of our interest and that use Internet to promote events. Therefore, Objectives 1 and 2 are covered.*

3.4 Event Information Extraction

At this point we have the organizations and their category. The next step is to identify the events that are contained in the web sites. To do this, first we identify the source of events and then we try to detect the meaning of each part of the information.

Figure 3.3 shows the event detection activity diagram of the process. The activity details can be found in Table 3.3.

ID	**Activity**	**Description**
1.1	Crawl Websites	The external websites are crawled at their second level. We assume that the information about the events and calendars is usually accessible via a depth of two from the home page.
1.2	Generate WARC	The HTML documents are stored in a WARC file.
1.3	Process WARC File	As in the previous process, this activity refers to extract the links and text from the HTML file and insert it into a graph.
1.4	Website analysis	With the resulting text and links, a Document-Term Matrix is created to identify the HTML documents that are likely to contain information about events. Also a model for filter the candidates is chosen. This model can be a SVM, CART model or any other method.
1.5	Candidate labeling	An expert should provide insight about the candidates to filter out those false positive and true negatives for the training model.
1.6	Pages filtering	Finalize the HTML documents identification that contains events.
1.7	Graph update	The HTML documents are updated into the graph.

Table 3.3: Event detection activities.

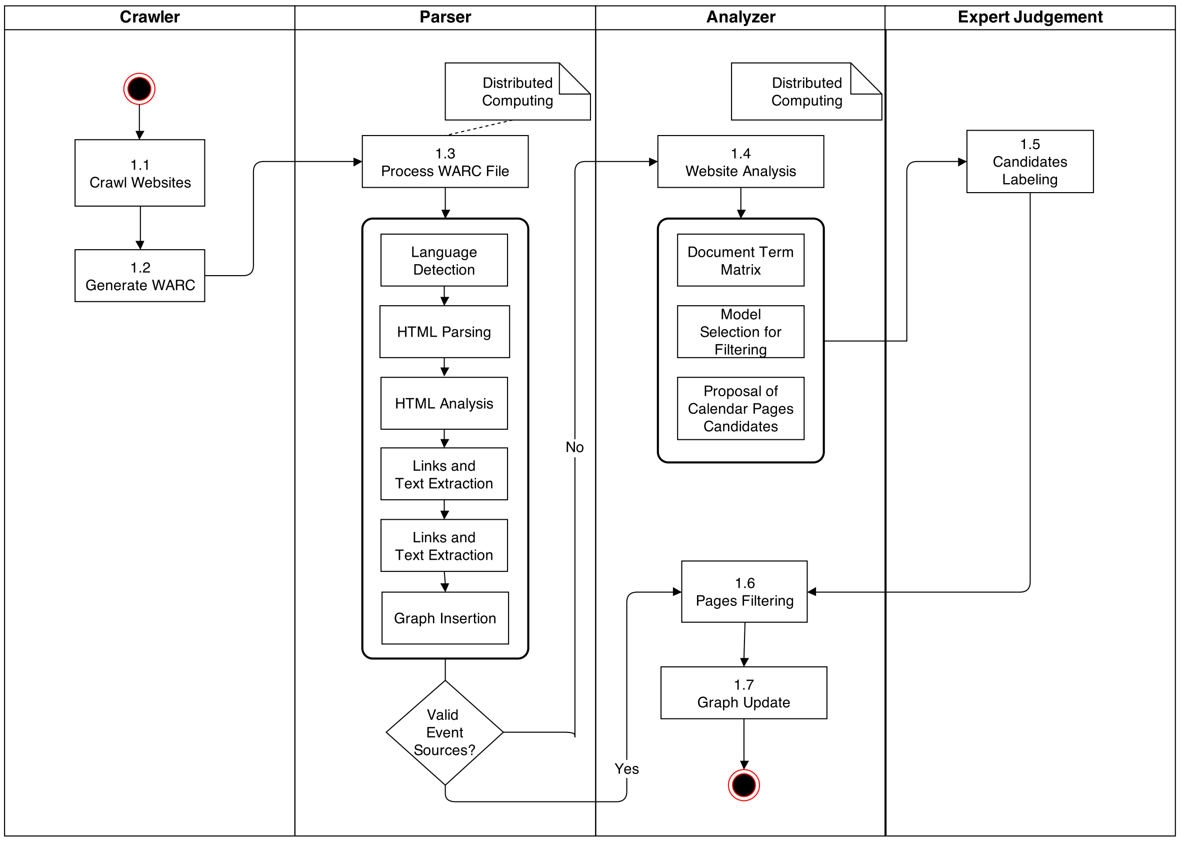


Figure 3.3: Event detection activity diagram.

Hypothesis 2. *By crawling the event organizers web site and performing the analysis of it, Figure 3.2 allows us to detect the sources of events for each organization. This covers Objective 3.*

Note that the expert insight is a key to validate that the information being processed is related to events. The objective of the system is to detect social events and we want to avoid as much as possible the identification of *fake events* and *fake positives* pages to extract events.

The second step is the event information extraction itself. The aim is to understand the meaning of each of the parts in the event. To achieve this, the system proposes a feature extractor and the expert validates it.

In other words, the feature extractor can be any algorithm able to identify and extract the pieces of event information. For example, the system can recommend one of the algorithms that we reviewed in the current research on event detection or any other. The feature extractor is used as an initial proposition and the expert decides if it is appropriate or it requires modifications.

Figure 3.4 presents the activity diagram of the event extraction process and Table 3.4 describes the details of the activities involved.

ID	Activity	Description
1.1	Get HTML documents of events	Retrieve the HTML documents marked as event pages.
1.2	Parse HTML documents	Extract the DOM representation of the HTML Documents.
1.3	Feature extractor recommendation	The system detects the appropriate feature extractor based on the DOM object.
1.4	Feature extractor validation	The expert validates if the feature extractor suitable for the group of web documents.
1.5	Feature extractor application	Extract the event information using the feature extractor.
1.6	NER detection	The aim of this task is to identify the category of the extracted features.
1.7	Validate extraction	The expert determines if the information extracted makes sense.
1.8	NER tuning	Improve the NER model with the new classes.
1.9	Graph update	The events information is updated in the graph.

Table 3.4: Event information extraction activities.

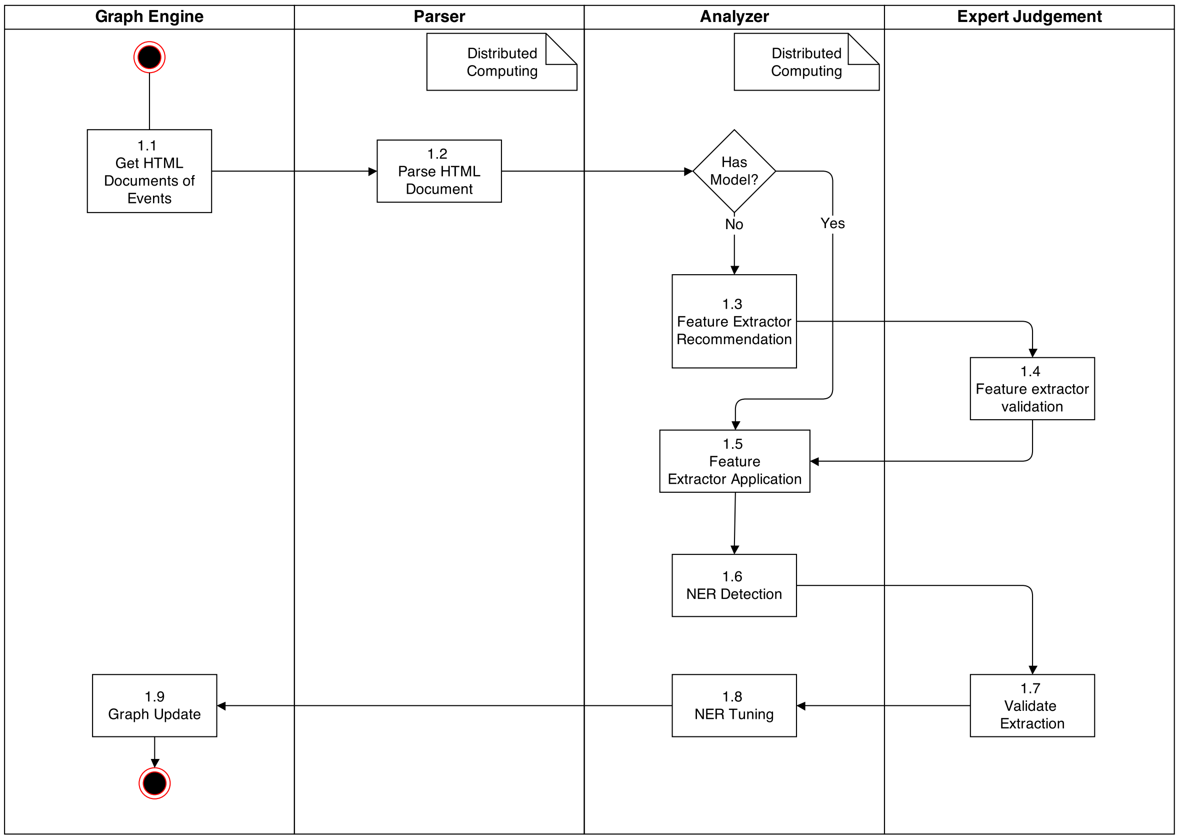


Figure 3.4: Event information extraction activity diagram.

The event information extraction process is related to the research objectives as the following hypothesis states.

Hypothesis 3. *Figure 3.4 allows us to extract the information of the event. gets the information of the event. Getting the required and optional event data and name the entities accordingly that is the Objective 4.*

Choosing the feature extractor is a crucial activity for the first event information extraction of one web site. We assume that the workload of the expert is very high during the first event extraction. Then, it decreases since web sites do not change their structure so often. Finally, it gradually increases again when the web sites change their known structure.

Another important activity is re-training the NER model. At the beginning, the NER model does not know many entities. However, the expert validates the extraction and corrects the named entities. This new information can be used to re-train the model.

3.5 Quality Improvement and Graph Construction

The very last step is to improve the quality of the data and connect all the events via the properties they have in common. Table 3.5 describes each of the activities and Figure 3.5 presents the activity diagram of the graph construction.

ID	Activity	Description
1.1	Get events per organization	Retrieve the detected events per organization.
1.2	Apply *R-Swoosh*	Run the *R-Swoosh* algorithm.
1.3	Identify duplicates	Detect those records that are likely to be duplicated.
1.4	Merge records	Combine duplicated records.
1.5	Update graph	Update or delete the corresponding nodes.
1.6	Analyze rules for new edges	The expert analyzes information to identify the relevant edges to be created (e.g. artists of the same genre, movies with same actors) with meta-data or the recognized named entities.
1.7	Define rules for new edges	The rules are defined in the analyzer.
1.8	Apply rules	The analyzer queries the graph to detect the edges to be created.
1.9	Update edges	Finally the graph is updated with the new edges.

Table 3.5: Quality improvement and graph construction activities.

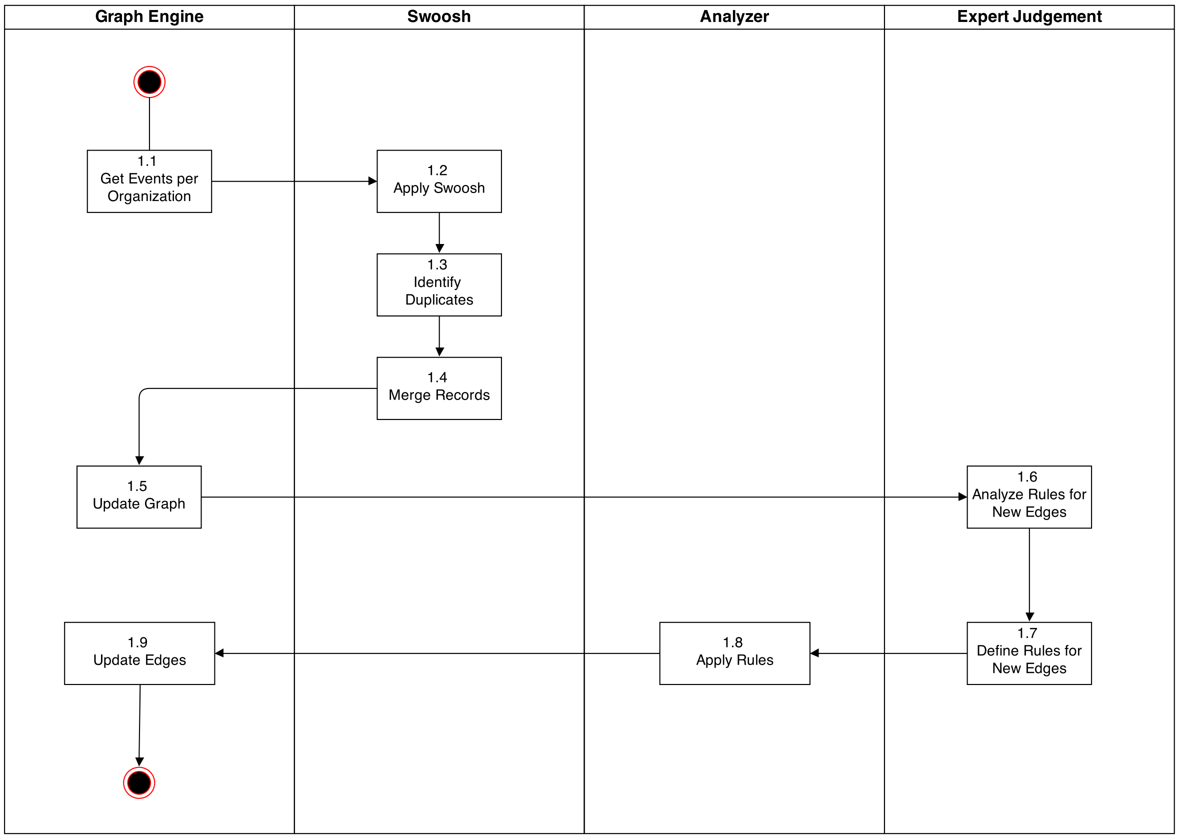


Figure 3.5: Quality improvement and graph construction sequence diagram.

The quality improvement and graph construction process is related to the research objectives as the following hypothesis states.

Hypothesis 4. *The steps in Figure 3.5 allows us to improve the quality of the data and create the event graph. Covering Objectives 5 and 6.*

The data quality improvement is a critical step to filter out information that is incomplete or duplicated to the user. The main benefit of this final step is that we fine tune the graph structure and create edges among the graph.

3.6 Development View

This view shows the system by its components to understand how the software components are organized. Figure 3.6 shows how small components can interact together to form bigger ones.

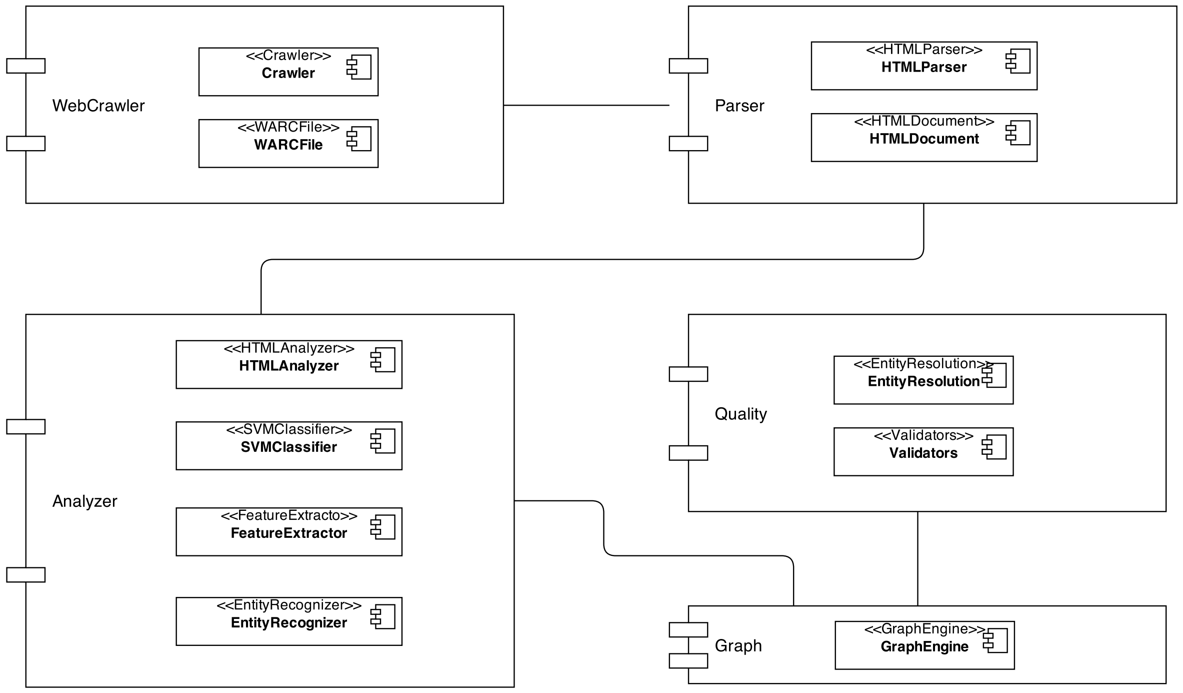


Figure 3.6: Component Diagram.

3.7 Physical View

This view is used to *map the software to the hardware* and target the non-functional requirements. We focus mainly in two aspects of the software. These are scalability and performance (throughput). Figure 3.7 shows the physical diagram for this view.

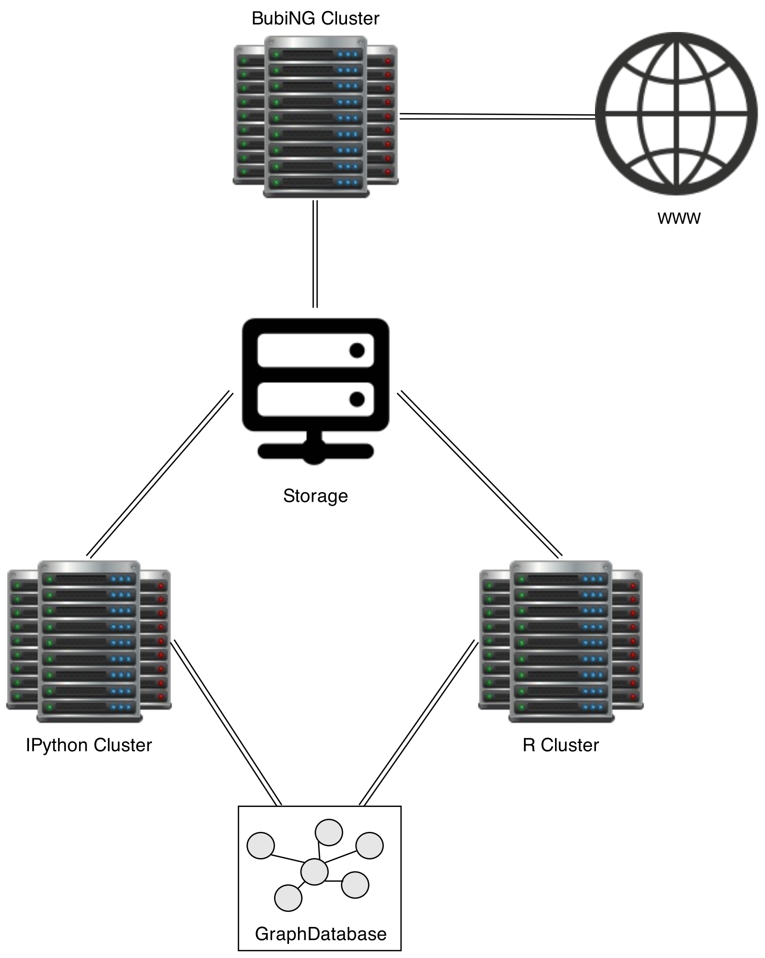


Figure 3.7: Physical diagram.

The physical diagram introduces some technologies that help us to achieve the non-functional requirements. These are BUbiNG Crawler [BMSV13], parallel computing with IPython [PG07] and R [Tea08].

The *BubiNG* [BMSV13] is a distributed open-source Java crawler. Table 3.6 describes some of its characteristics.

Characteristic	Description
Distributed	Multiple agents crawl the web independently from a central control that increases the crawling speed as the number of agents are increased.
Configurable	Various parameters and rules for crawling, fetching, storing can be set on the start-up and changed while crawling using JMX.
Hardware Efficient	The crawler is designed to take advantage of the multi-core processors and RAM memory.
Guarantees Politeness	It respects request intervals for crawling the host and IP addresses

Table 3.6: Some BubiNG characteristics.

3.7.1 Parallel Computing in Python and R

Python is a high-level and general-purpose programming language that has several modules for text processing, data structures, graphs and data analytics. We would like to use these libraries but we need to be sure that it is possible to distribute the work-load among different processes.

IPython [PG07] is an interactive shell for Python. We are mainly interested in the parallel computing features it provides. These features enable the creation of single program multiple data parallelism, multiple program multiple data parallelism or any other custom combination the user would like to do.

IPython uses an engine and a controller. The engine is in charge of taking the commands to Python shells that can be distributed over the network. The controller is an interface that interacts with the engines and contains a hub and a scheduler. The hub keeps track of engine connections, schedulers, clients, as well as all task requests and results. The scheduler provides a fully asynchronous interface to a set of engines [RK07].

R is a very powerful language designed to solve statistical and mathematical problems [Tea08]. There are special graph and data mining libraries for R. By using different *apply* functions or special packages like *RScaLAPACK* or *rmpi* it is possible to distribute and parallelize the tasks in R [NFSC13].

3.8 Scenarios

We now review the interaction between the user and the system in the use case of Figure 3.8.

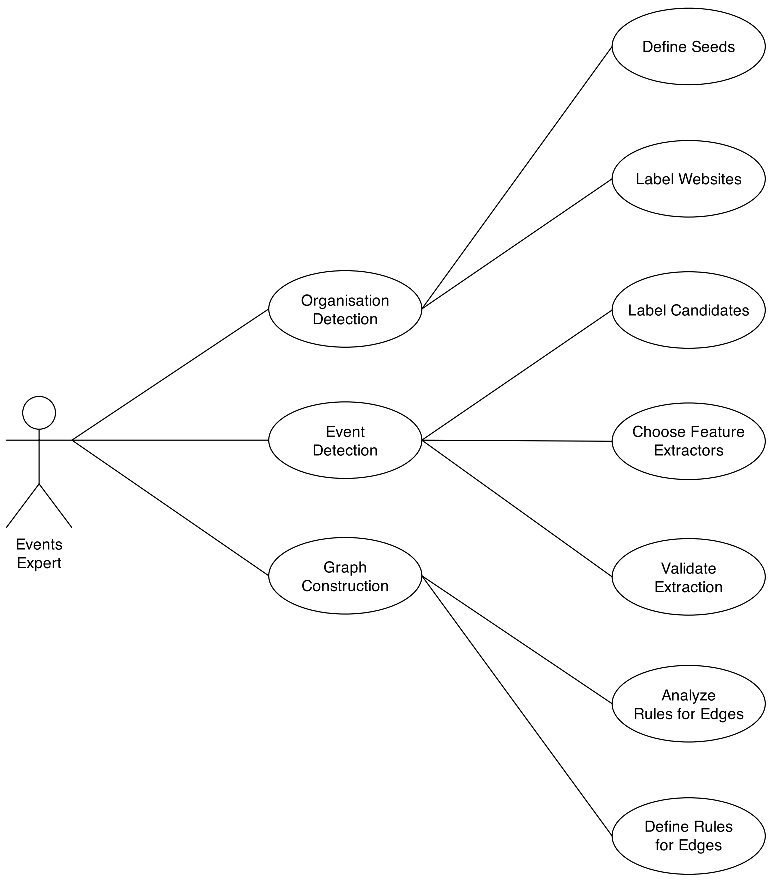


Figure 3.8: Use case diagram.

Structuring events data from the web requires the user to be actively involved in the process. This is because processes require information that is not available in the system domain at the beginning. Furthermore, the classifiers and feature extractors requires input and feedback from the user to improve the outcomes of the process.

In this chapter we reviewed the software architecture of a system using different views. The logical view showed the functionality that the system provides to the end-user. The process view presented 4 processes that cover the thesis objectives. The development view helped us to understand the system by its components and how they are organized. In the physical we presented technologies that can run the software and cover the non-functional requirements. Finally, in the scenarios we reviewed the interaction between the user and the system. The next chapter presents an experiment to validate the software architecture in a real case.

Chapter 4

Experiment and Results

This chapter shows the design and results of an experiment that implements part of the event detection system for a very specific location and organization type. The chapter is organized as follows. First, the experiment is presented, followed by the results of the event detection process. Finally, we compare the results of the experiment with other sources of social events.

4.1 Experiment Design

The objective of this experiment is to identify if the event detection system is usable in a real life case. To do this we follow the processes described in the software architecture.

Social events are geographically bounded. Thus, we geographically limit the search. We focus on social events happening in Budapest, the capital and largest city of Hungary.

For the experiment, two seeds for organization detection are considered. These are budapest.com and budapestinfo.hu. Both of them allows web crawling since there are no policies on the *Robot Exclusion Standard* file. This file is usually a *robots.txt* file that contains the rules to allow or deny permission to the resources of the web site. Additionally, no content besides the redirection to external websites is used in this thesis work.

Table 4.1 introduces the steps performed in the experiment.

Step	**Description**
Organization detection	Based on the given seeds detect the concert halls, theaters, museums, galleries, festivals, nightlife and tours.
Concert halls events detection	Detect the events happening on the concert halls detected in the previous objective.
Event information extraction	Extract the pieces of information of the detected events in the previous step.
Entity resolution	Improve the quality of the social events detected.
Social events graph construction	Build the network of the social events detected and create the edges between nodes that are missing. In this step additional meta-data can be added from third parties to improve the network structure.

Table 4.1: Experiment steps.

4.2 Results

4.2.1 Organization Detection

After crawling and processing the WARC files of each seed, we have a graph of the documents of each seed. Since we do not know which of the HTML pages are relevant we rank them by calculating centrality measures. In fact, both HITS and PageRank proved helpful to rank the HTML documents. By sorting these documents by their rank it is easy to detect the documents for the organization detection process. This part of the experiment is related to Hypothesis 1.

Figure 4.1 shows the HTML documents of the seed budapestinfo.hu sorted by PageRank. Note that the main (i.e. left) axis measures the hubs and authorities while the secondary (i.e. right) axis measures PageRank. We can observe a cut around the document 45. Among the documents ranked as relevant, we find ones related to culture, gastronomy, free-time, walks, tours, churches and memorials, museums, galleries and music.

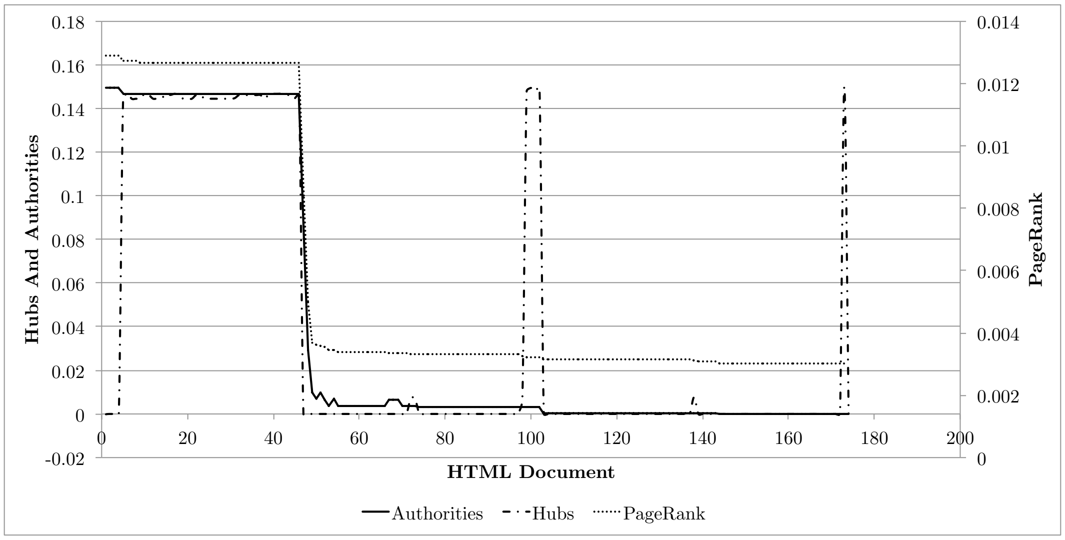


Figure 4.1: Seeds HTML documents centrality measures.

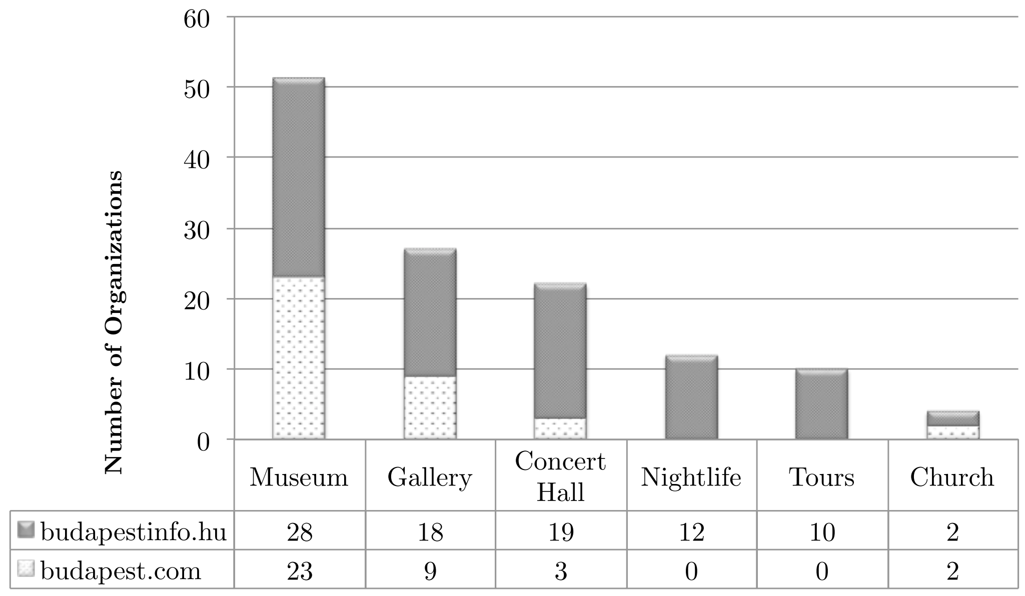
In this particular case, there is no need to train a SVM to classify the pages. This is because labeling the organization categories in the given sources is a simple task. For example, the HTML document *budapestinfo.hu/museums.html* can be easily labeled as a source for museums, avoiding false positives.

However, it is required to filter out the common social networks (e.g. Facebook, Twitter, Tumblr) and the frequent external links. Table 4.2 shows the most frequent external links. Since the difference between the average and the higher frequent external link is big, it is easy to filter them.

External Domain	Frequency
h-i.hu	7545
debrecencity.com	5025
budapest-hotel-guide.hu	18
bfnp.hu	18
town-hall-apartment-budapest.com	18
danubiushotels.com	10
cinemacity.hu	9

Table 4.2: Top 7 count of links to external sites sorted in decreasing order.

By following the activity sequence presented in Figure 3.2 we identify 126 possible event organizer candidates. We find 51 museums, 27 galleries, 22 concert halls, 12 nightlife, 10 tours and 4 churches. In Figure 4.2 the sources of each of the organizations is shown.



	Museum	Gallery	Concert Hall	Nightlife	Tours	Church
▨ budapestinfo.hu	28	18	19	12	10	2
▨ budapest.com	23	9	3	0	0	2

Figure 4.2: Organizations detected per seed.

In Figure 4.3 a graph of the organization detection process is shown. We see that some organizations (i.e. smaller in size) are referenced by more than one seed. It turns out that we have 110 distinct organizations out of the 126. The difference is because seeds can link to the same organization and because organizations may have more than one category.

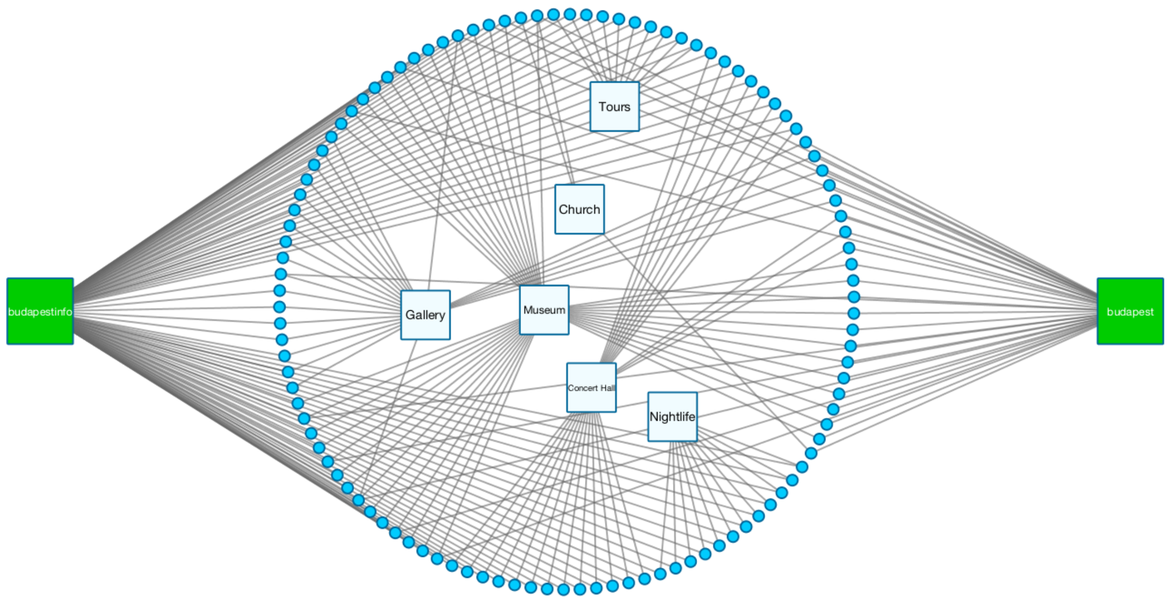


Figure 4.3: Graph of the detected organizations.

4.2.2 Concert Halls Event Detection

To perform the event detection and the event information extraction we use the concert halls detected in the previous steps. Although detecting and extracting information of concerts is a challenging task, there are several sources of information where we can benchmark the results of the process. In particular, Last.fm, Eventful and Facebook.

Table 4.3 shows the concert halls detected in the process of organizations detection we reviewed previously.

Web Site	Organization	Availability
bcwtc.hu	Budapest Congress Center	Available.
matyas-templom.hu	Matthias-Church	Available.
nagyhazi.hu	Nagyházi Gallery	Not Available.
corvinteto.hu	Corvintető	Available.
budapestarena.hu	Budapest Arena	Available.
millenaris.hu	Millenáris	Available.
merlinbudapest.org.hu	Merlin Budapest	Not Available.
bcc.hu	Budapest Congress Center	Available.
godorklub.hu	Gödör Klub	Available.
a38.hu	A38	Available.
bof.hu	Café Budaest	Available.
mupa.hu	Palace of Arts Budapest	Available.
lfze.hu	Liszt Academy	Available.
trafo.hu	Trafó	Available.
iicbudapest.esteri.it	Istituto Italiano di Cultura	Available.
obudaitarsaskor.hu	Óbudai Társaskör	Available.
dancetheatre.hu	National Dance Theatre	Available.
fono.hu	Fonó	Available.
dunapalota.hu	Duna Palota	Available.
hagyomanyokhaza.hu	Hungarian Heritage House	Available.
szabadter.hu	Open-air Theatre Kht.	Available.
operettszinhaz.hu	Operetta Theatre	Available.

Table 4.3: Concert halls detected.

It happens that the same organization can have multiple domains. That is the case for the *Budapest Congress Center* which has the domains *bcwtc.hu* and *bcc.hu*. Also, not all the web sites are available by the time of the crawling.

After crawling the websites to the second level of depth and processing the resulting WARC files, we can select the HTML documents candidates that are used for event detection and event information extraction. This part of the experiment is related to Hypothesis 2 and Hypothesis 3.

The detection of candidates is based on three characteristics of the HTML documents. First, only documents in English are processed. Second, we check if the document contains meta-data about event using the schema.org/Event item-type. Finally, we parse the document counting the dates greater than the day of the document parsing and compare it to the other HTML documents.

Figure 4.4 shows the recommendation for labeling the HTML documents as sources of events. The experiment succeed in identify the sources of events based on the three proposed features.

The next step in the activity process is to manually validate the candidates as event sources of information and store them in the graph. Table 4.4 shows the event sources for each of the website.

Organization	Source of Events
A38	a38.hu/en/programs
Budapest Arena	en.budapestarena.hu/html/kovetkezo_esemenyek.html
Budapest Congress Center	bcc.hu/en/public-events.html
Corvintető	corvinteto.hu/en/programs
Fonó	fono.hu/programs
Hungarian Heritage House	heritagehouse.hu/main/performances/
Liszt Academy	lfze.hu/en/all-programs
National Dance Theatre	dancetheatre.hu/esemenynaptar/esemeny/
Óbudai Társaskör	obudaitarsaskor.hu/index.php
Open-air Theatre Kht.	eng.szabadter.hu/
Operetta Theatre	operett.hu/operett.php?pid=schedule
Palace of Arts Budapest	mupa.hu/en
Trafó	trafo.hu/en-US/programs

Table 4.4: Final candidates for the web sites.

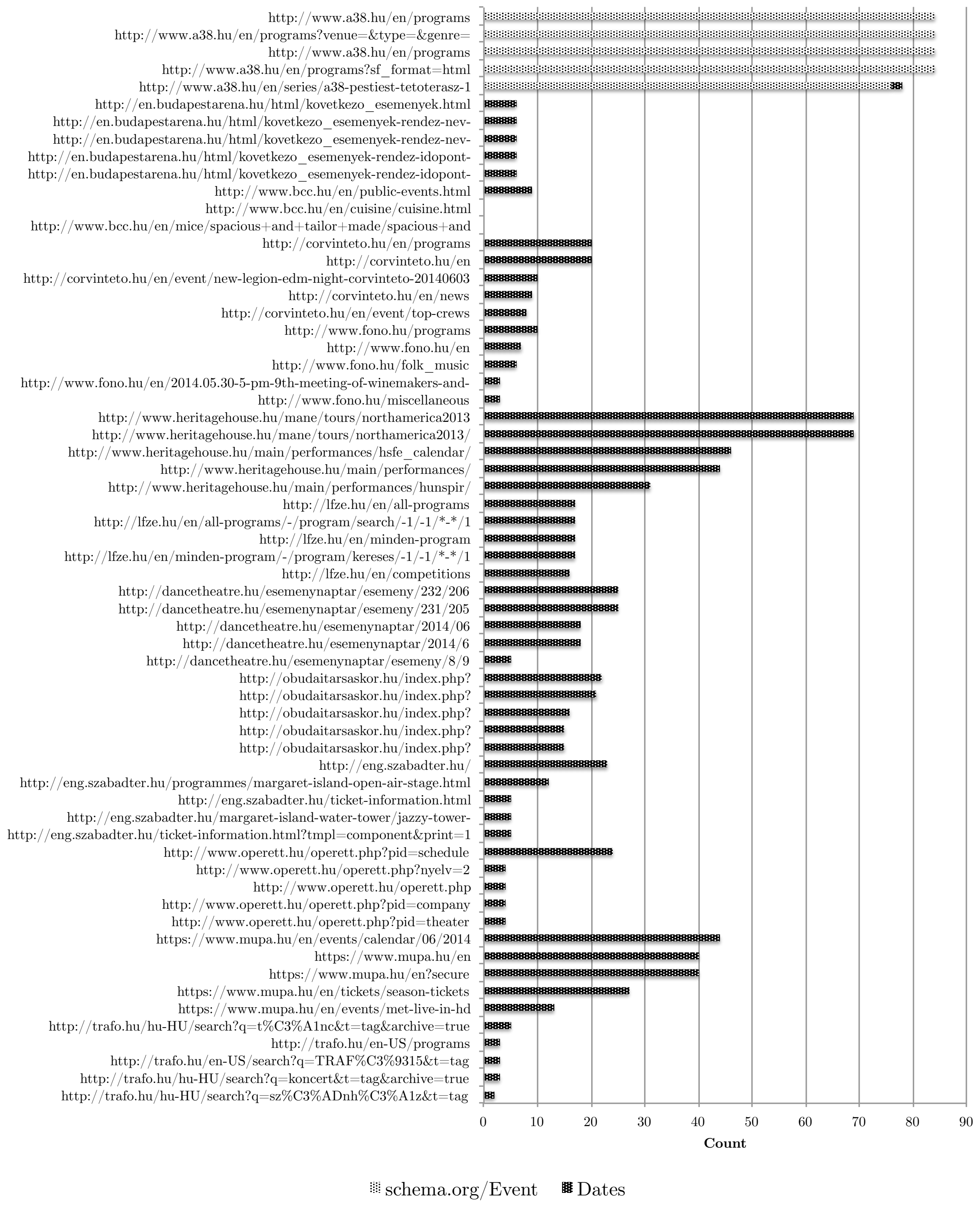


Figure 4.4: Top 5 HTML documents candidates for event extraction.

Once we have the event sources defined, our task is to extract the pieces of event information. First, we review the feature extractor designed for the experiment. After that, an example of the application of NER technique is presented. Finally, ER is applied in the events data and the the social events graph is created.

The feature extractor recommendation implemented in this experiment is based in analyzing the HTML document structure. The process analyze how common are the CSS classes, element ID and the HTML structures itself among the dates founded in the previous process.

Based on the frequency of these characteristics, the system presents a suggestion and the expert decides to accept or modify the feature extractor. Finally, we use a specialized HTML parser [Nai14] to extract these features. The following is a detailed explanation of how this algorithm works.

The first step for the is to get the last version of the HTML file from the event source. Afterwards, we analyze the structure of it by looking for dates in the HTML file. For each HTML element that contains the date, we analyze the path from this element to the body. The structure is stored as a string representing the HTML element, the CSS class and the ID. Figure 4.5 is an example of the path from the HTML document to the element containing a date. The square brackets are used to show the CSS classes while the curly brackets contain the IDs.

By doing this for all the dates in the HTML document, we identify the frequency of each part of the structure. Thus, we can infer the HTML element that contains the events. The example is taken from the event source of *Corvintető*. It turns out that the structure is common among the 18 dates. In Figure 4.5 this structure is filled with gray.

The next step would be to apply the same process in the HTML element that contains the events. If we check the *div[view-content]* element, it indeed contains the information of the events. Then, by performing the same structure analysis, we identify the classes in Table 4.5. These classes represent the information of the event.

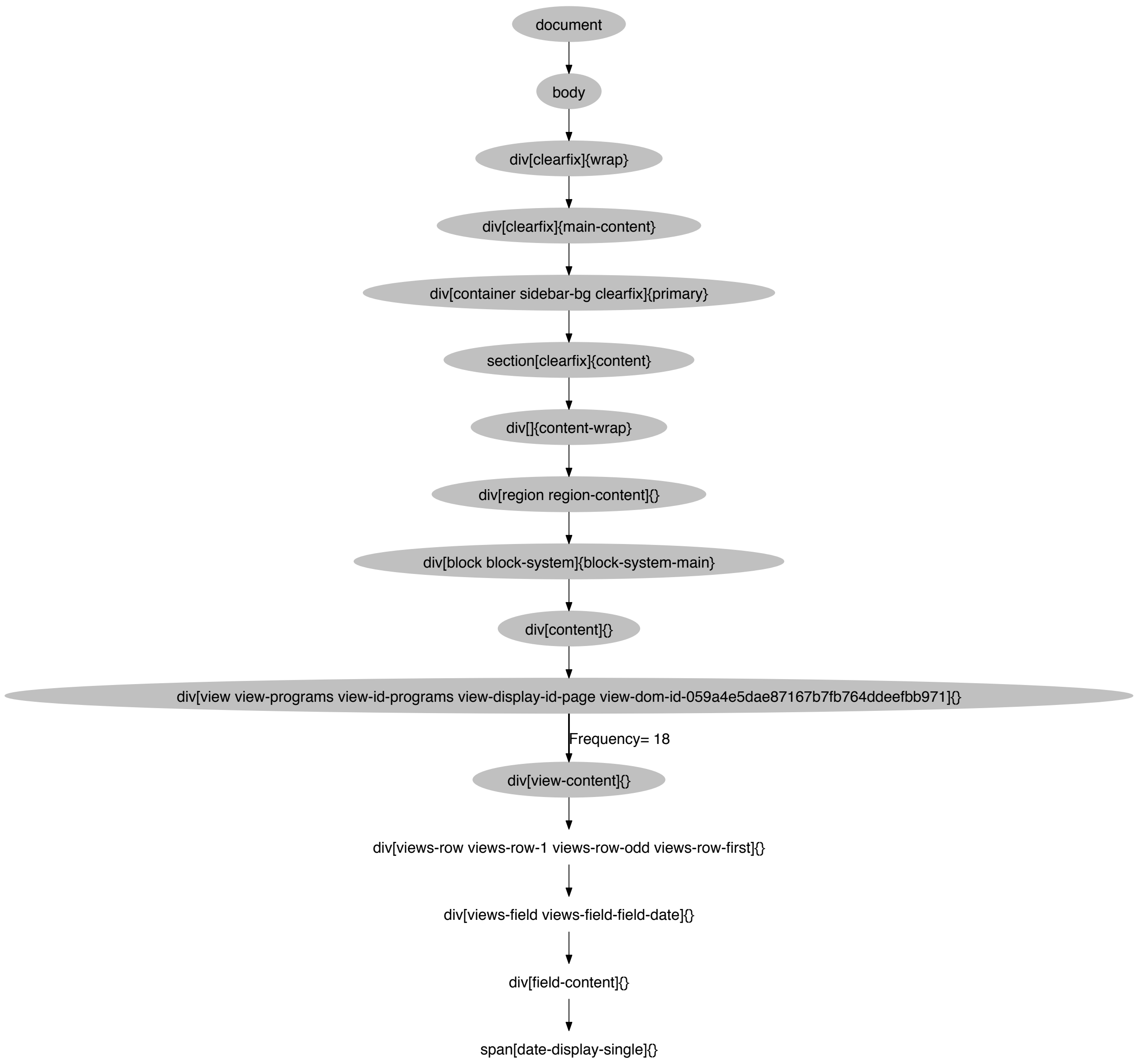

document
body
div[clearfix]{wrap}
div[clearfix]{main-content}
div[container sidebar-bg clearfix]{primary}
section[clearfix]{content}
div[]{content-wrap}
div[region region-content]{}
div[block block-system]{block-system-main}
div[content]{}
div[view view-programs view-id-programs view-display-id-page view-dom-id-059a4e5dae87167b7fb764ddeefbb971]{}
Frequency= 18
div[view-content]{}
div[views-row views-row-1 views-row-odd views-row-first]{}
div[views-field views-field-field-date]{}
div[field-content]{}
span[date-display-single]{}

Figure 4.5: Example of Analysis Structure.

CSS Class	**Content**
views-field-field-image	Link to image
views-field-title	Link to HTML document
views-field-field-date	Date
views-field-field-kapunyitas	Time
views-field-field-jegyar	Text
views-field-nothing	Empty

Table 4.5: Feature extractor recommendation.

For this example, the expert judgment is to remove the string contained in the class *views-field views-field-nothing*. And label the classes as shown in Table 4.6.

Class	Content
views-field-field-image	Event Image
views-field-title	Event Title
views-field-field-title	Event Website
views-field-field-date	Date and Time
views-field-field-kapunyitas	Doors Open
views-field-field-jegyar	Ticket Price
views-field-nothing	*Ignore*

Table 4.6: Expert Judgment on the proposed feature extractors.

Figure 4.6 shows the number of events detected per source. Note that in the case of operett.hu it is not possible to extract the events since the HTML document does not contain a pattern in class, ID or structure.

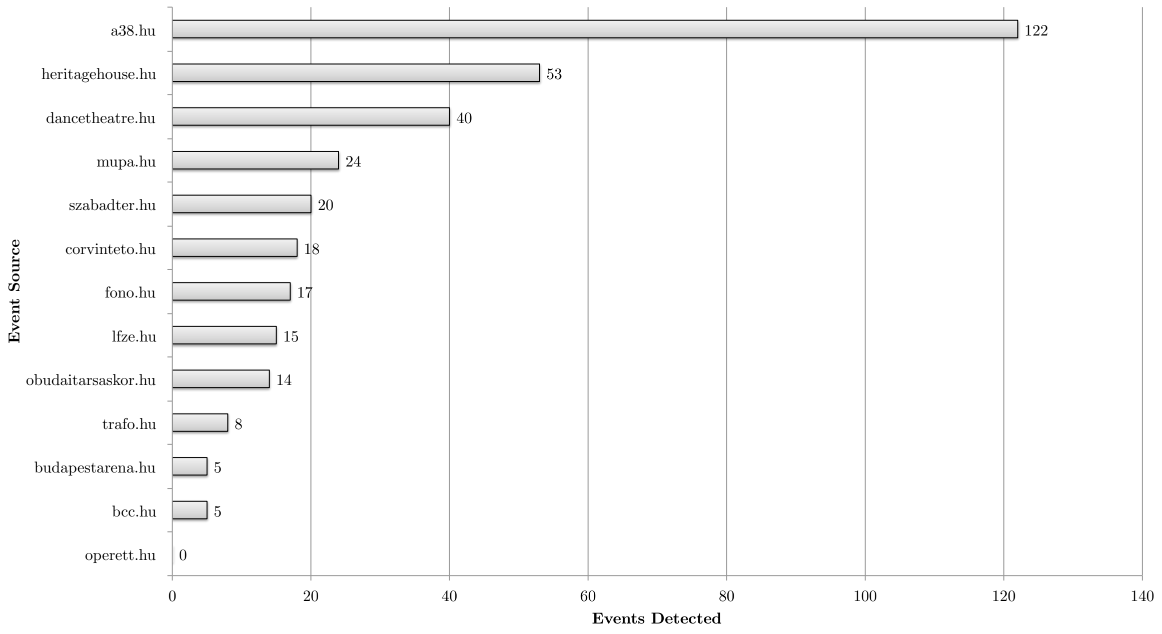
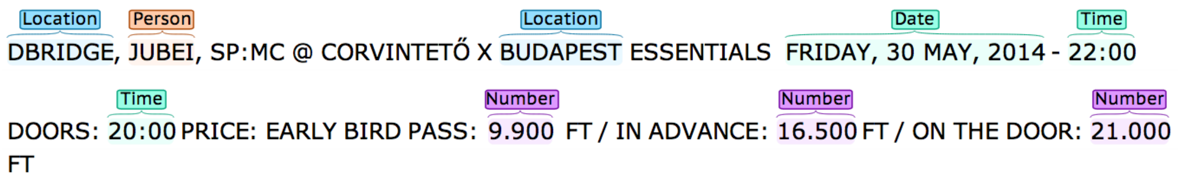


Figure 4.6: Events detected per source.

Once the information is labeled, we can proceed to extract the information of all the events and apply the NER process on the extracted data. Figure 4.7 shows the result of running the NER process with the default trained model provided by the Stanford CoreNLP System, particularly the NER annotator [FGM05].


DBRIDGE, JUBEI, SP:MC @ CORVINTETŐ X BUDAPEST ESSENTIALS FRIDAY, 30 MAY, 2014 - 22:00

DOORS: 20:00 PRICE: EARLY BIRD PASS: 9.900 FT / IN ADVANCE: 16.500 FT / ON THE DOOR: 21.000 FT

Figure 4.7: NER Example of an event.

The result of the first NER successfully detect the date, time and ticket price. The algorithm did not perform well for labeling the artists and currency. These labels are manually detected by analyzing the parts of the HTML file that represents artists, genre and event type. The extracted labels are shown in Table 4.7.

Text	Entity
DBRIDGE	Artist
JUBEI	Artist
SP:MC	Artist
Electronic	Genre
Concert	Event Type

Table 4.7: Corrected and Added Entities.

The benefit of NER is that the model can be trained over time. After each iteration in the entity recognition process, it is possible to re-train the model with the additional text and entities detected. Since performing the re-training and validation of the NER model can be an extensive task, this is not included in the experiment. Further work can include more extensive test in the re-training and evaluation of the NER annotator.

The last part of the experiment is to improve the quality of the data and create the graph of social events. This part of the experiment is related to Hypothesis 4.

For this experiment the ER implementation provided by the Standford Entity Resolution Framework [BGMM$^+$09] is used. This process uses an XML file containing a register for each of the events and their attributes. These are *Event Title*, *Metadata*,*URL* and *Entity Type*.

The number of XML input records was *788*. This data is used as input for the *RSwoosh* algorithm. If the program returns the same number of registers it means that there are no duplicated information. This is the case in our experiment.

Finally, with all the information from the previous processes, the social events graph is created. The process starts by creating a node for each of the dates, event and their meta-data (i.e. artist, movie, event type and music genre). Then, the relations among these nodes are created.

Figure 4.8 presents the different nodes grouped by their type. The circles are sorted by the number of out-going edges. Some nodes are more relevant in the network because of the amount of out-going and in-going edges. We can observe that some organizations, artists, genres, events, event types and dates are more connected to other nodes. Note that this structure can be used in the future to perform graph mining in this network.

Figure 4.9 is another representation of the same graph.

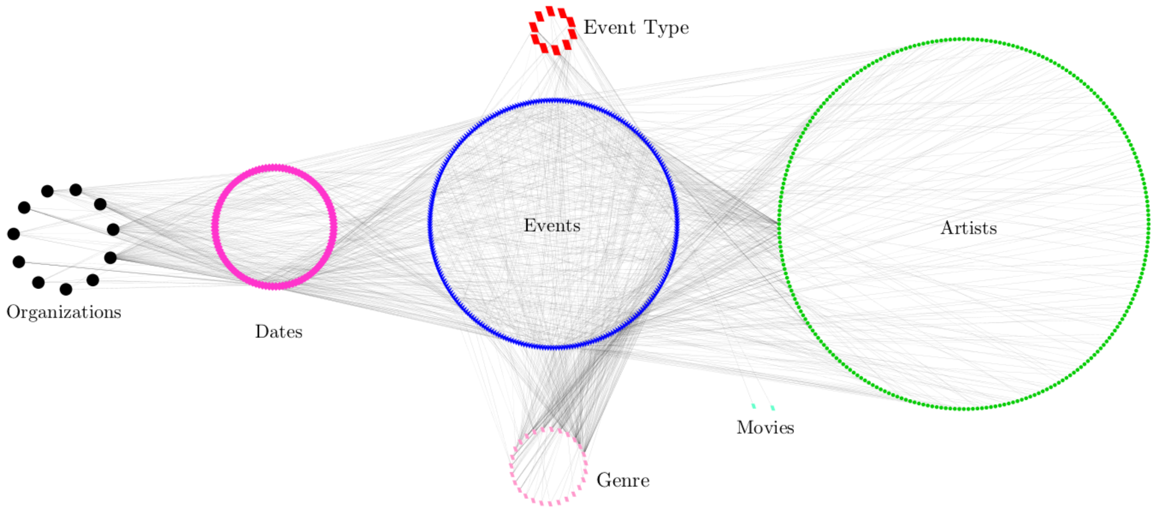


Figure 4.8: The social event graph per group sorted by degree.

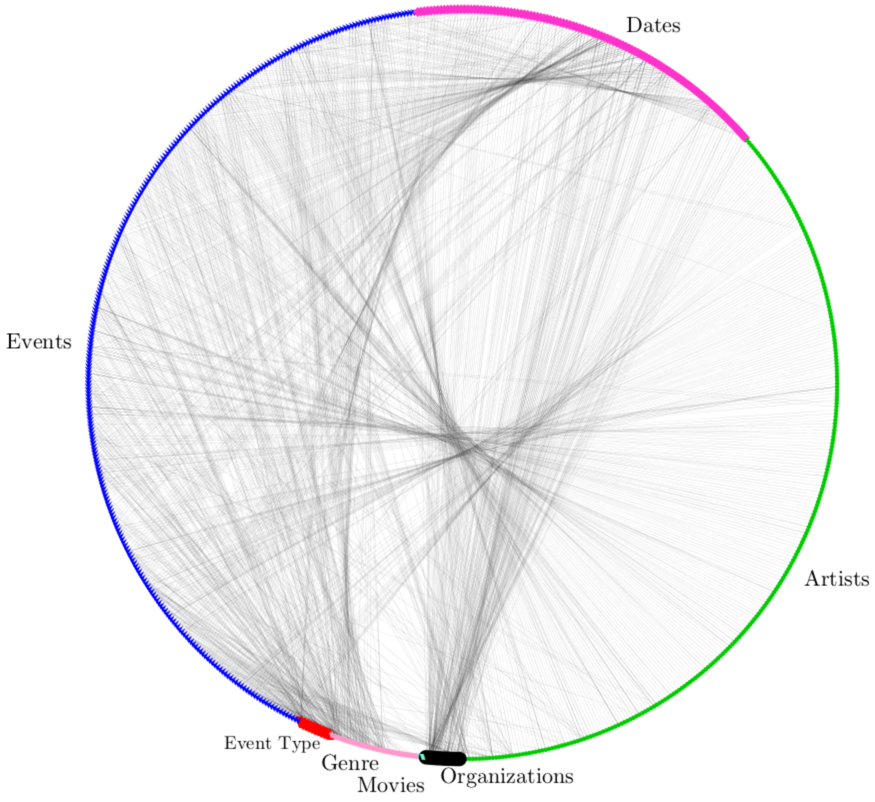


Figure 4.9: Circle representation of the social events graph.

4.3 Comparisons

To benchmark the results of the experiment, we use some of the most popular social networks. These may have information about the events of the concert halls we analyzed. This section starts by introducing each of the selected social networks, followed by the reason why it is selected for the benchmark. Then, a comparison between the experiment results and the social networks events data is presented.

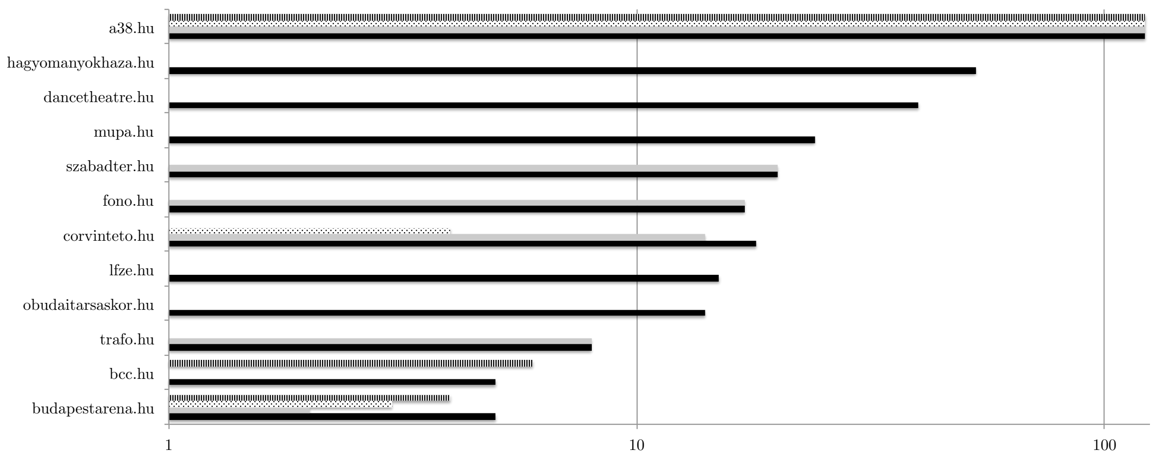
Facebook is the biggest social network in the world. It has 1.28 billion monthly active users by March 2014 [Fac14]. It offers to individuals the possibility to join this network for free by registering an account. This allows them to find friends, share photos, create a profile page, follow pages, like content, organize events, play games and so on. Facebook offers also the possibility for public and private organizations to create a Facebook page. With these pages organizations can provide information to the users. The information can vary from photos, videos, text, locations and events. We are particularly interested in retrieving the events information of the official concert hall Facebook pages to compare it with our experiment results.

Last.fm is a social network focused on music. Users can register an account for free. Some of the features that Last.fm offers are profiles, recommendations, groups and events. Any user can add a new event or concert hall and also they can link the artists that perform in the events. We retrieve the events information on the Last.fm concert halls pages and compare it to the results of this thesis.

Eventful is a social network focused on events. This organization gathers the data of several event organizers. This enables the users to navigate the content based on 29 different categories. Users can also search events by location, artist or concert hall. To compare our results with Eventful we retrieve the events information related to the concert halls considered in the experiment.

The first step of the comparison is to retrieve the events information of the three social networks. Then, we compare the events per concert hall. The events are compared by checking the date, time and event title.

In Figure 4.10 we can see the number of events detected. Figure 4.11 shows the same rectangle size when the amount of events detected are equal. For example, in the case of *a38.hu* the social networks and the experiment contained the same events information. Thus, the rectangle size is the same. That is not the case for *budapestarena.hu* where the rectangles size is different.



	budapestarena.hu	bcc.hu	trafo.hu	obudaitarsaskor.hu	lfze.hu	corvinteto.hu	fono.hu	szabadter.hu	mupa.hu	dancetheatre.hu	hagyomanyokhaza.hu	a38.hu
▥ Eventful	4	6	0	0	0	0	0	0	0	0	0	122
░ Last.fm	3	0	0	0	0	4	0	0	0	0	0	122
■ Facebook	2	0	8	0	0	14	17	20	0	0	1	122
■ Experiment	5	5	8	14	15	18	17	20	24	40	53	122

Figure 4.10: Number of events per concert hall and source.

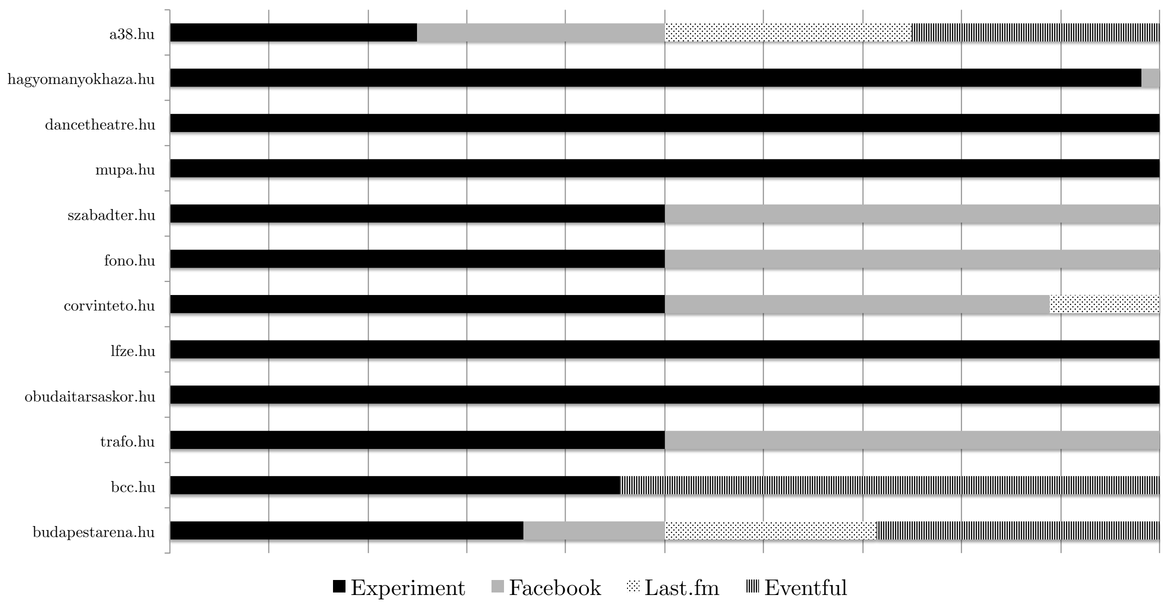


Figure 4.11: Similarity of detected events vs registered events in the social networks.

Note that for *a38.hu*, *bcc.hu*, *budapestarena.hu*, *corvinteto.hu*, *fono.hu*, *szabadter.hu* and *trafo.hu* there is similarity between the number of detected events and events registered in the social networks. This information validates that the proposed software architecture works and is able to extract event data from the web. In the case of *dancetheatre.hu*,*hagyomanyokhaza.hu*,*lfze.hu*, *mupa.hu* and *obudaitarsaskor.hu* there is no point of comparison but we see that the proposed software architecture can complement the information in social networks.

In this chapter the results of the conducted experiment are presented showing the validity of the processes described in the previous chapter. It is valid because the steps in the experiment were successful. The organization detection process identified 110 event organizers. Then, a process crawled the websites of the concert halls and applied an algorithm that detected the HTML document for event extraction. A feature extractor was proposed based on a HTML algorithm that analyzes the common CSS classes, IDs and HTML structure. The features were presented to the expert who decided if those features were right or required modifications. The NER process was executed with the extracted information as input. Few entities were recognized and the expert provided insight on the right entities labeling. Afterwards, the ER method did not detect duplicates among the events. Finally, a social events network was created using the information from the previous processes. The next chapter is a conclusion of this thesis and concludes the together with ideas for future work, limitations and weaknesses of this thesis.

Chapter 5

Conclusion

This chapter concludes the thesis. To do so, the first section presents a summary of each of the chapters. Then, a general conclusion and contributions of this work are presented. Afterwards, a description of the possible future work in the direction of structuring social events is presented. Finally, the limitations are presented.

5.1 Summary of the Chapters

The first chapter states the research problem by giving an introduction to the Internet as a marketing tool for event organizers and explaining how promoting events on Internet brings a challenge to the user because of the data distribution. Then we reviewed the current research on event detection, paying attention in those who use general text streams (e.g. news, articles), social media streams (e.g. Twitter) and multimedia (e.g. Flickr) to detect events. Finally, the research objectives were presented. These are summarized in Table 5.1

Objective	**Description**
Objective 1	Identify the event organizers that fit in at least one category of our interest and that use Internet to promote events.
Objective 2	Based on reputable web sites (e.g. official tourist websites) we detect the web sites of organizations that promote events.
Objective 3	Identification of those parts of a HTML document that might describe an event.
Objective 4	Extract the required and optional event's data and name the entities accordingly (e.g. artist, location, ticket price).
Objective 5	Enhance the data quality of event detection to improve the outcomes of the process.
Objective 6	Create a graph of social events using the resulting information of the previous objectives. Complement the resulting graph with additional information available on the Internet.

Table 5.1: Research objectives summary.

Chapter 2 introduced the theoretical background required for the construction of the social events graph. We reviewed several concepts of graphs such as nodes, edges, degree, types of graphs and different ways to represented them. Then, two graph link analysis techniques (Hiperlink-Induced Topic Search and PageRank) were introduced as methods to understand the graph structure. Afterwards, three classification algorithms, logistic linear regression, CART and SVM were described. The chapter ends with the concepts of NER to identify entities and ER to improve data quality.

In chapter 3 the main contribution of this thesis work is presented. That is, a software architecture of a system able to satisfy the objectives of this research. To describe the architecture different views were shown. The logical view showed the functionality that the system provides to the end-user. The process view described four processes linked to the research objectives. The development view presented the system by its components to understand how the software components are organized. Finally, the physical view *mapped the software to the hardware* and proposed satisfying the non-functional requirements by using BUbiNG Crawler [BMSV13], parallel computing with IPython [PG07] and R [Tea08].

In chapter 4 an experiment to validate the ideas presented in the software architecture was executed. The objective of this experiment was to identify if the event detection system is usable in a real life case. The experiment was bounded to detecting event organizers in Budapest, the capital and largest city of Hungary.

For the organization detection process there were two seeds. These were budapest.com and budapestinfo.hu. After crawling them, the application of HITS and PageRank resulted in the identification of the most relevant documents of each seed. Then, the expert manually labeled them and it was possible to identify 126 event organizers candidates. Out of them 51 museums, 27 galleries, 22 concert halls, 12 nightlife, 10 tours and 4 churches. However, some of these organizations are referenced by the seeds to out of the 126 organizations, 110 are unique.

For the second part of the experiment we focused only in the detected concert halls. A process crawled the websites of the concert halls and applied an algorithm that detected three features. First, only documents in English are processed. Second, we check if the document contained meta-data about events using the schema.org/Event item-type. Third, we parse the document counting for the dates greater than the day of the document parsing and compared it to the other HTML documents.

Once the event sources were detected, a feature extractor was proposed based on a HTML algorithm that analyzes the common CSS classes, IDs and HTML structure. The features were presented to the expert who decided if those features were right or required modifications.

Afterwards, the information was extracted based on the chosen feature extractor. With this information, the NER process was executed. We observed that at the beginning very few entities were recognized. This is because the original training set of the NER model is not suitable for the particularities of concerts. To solve this problem, the expert provided insight on the right entity labeling.

The final part of the experiment was to apply the ER method to improved data quality. We found no duplicated information. This is likely because there are few entities in common among the concert halls. Finally, a social events network was created using the information from the previous processes.

Based on this we can conclude that the hypotheses were satisfied by following the proposed activities diagrams.

5.2 Contribution

First, the inclusion of the concept of trust in the event detection process. It turns out that trust is an important factor in detecting event organizers by using reputable seeds. Additionally, the classification algorithms are able to predict the class of event organizers and enhance the filtering of possible fake event organizers.

Moreover, we proposed a process to detect the event source of information inside a website. Additionally, a feature extractor is included to enable event information extraction. Then, NER provided a way to understand better what the information of the event means. Finally, the quality is improved using ER and the information of the social events is structured as a graph.

The proposed system provides enough flexibility to deal with different algorithms according to the user needs. Additionally, the non-functional requirements such as scalability and performance are considered by proposing technologies able to deal with these requirements.

Additionally it was shown that not all the information about social events are on the popular social networks. This is because the social networks and the official web pages of the events organizers can be out of synchronization. Thus, showing different content to the user.

Finally, by representing the structure of the social events, it is possible to create innovative ways to enable the users to navigate this information. For example, using data analytics, mash-ups and APIs to help the user to understand the social events information.

5.3 Possible Future Work

Further work can include a more diverse consideration of languages, cultures and organizations types. Additionally, the algorithm that parses the dates can play a key role for event detection, in the experiment we observed that some HTML documents contains the date in different elements.

In relation with the feature extraction, analyzing the HTML can be a great way to solve the event information extraction process. A deeper study on sub-pattern matching in HTML DOM structures can help to improve this process by taking the advantages that a marked up language offers (e.g. elements, IDs, classes).

Also, further work can be focused particularly on the NER process to compare different annotators and training data sets. The benefit that this can bring in the future is to reduce the amount of manually annotated data from the expert. The same apply for the ER process. Further research should be performed to analyze how the system behaves when there are cases that require data improvement.

The social events graph can be analyzed using graph mining techniques. Also, human- computer interaction studies can help to develop a system that allows the user to navigate this information.

5.4 Limitations

Table 5.2 shows the limitations for the experiment in relation to the hypotheses and activity diagrams.

Hypothesis	**Activity Diagram**	**Limitation**
1	3.2	The experiment did not use any classification algorithm (e.g. SVM). Thus, that part of the activity diagram was not needed to test.
2	3.3	The experiment considered only English language and the HTML structure of some event sources caused problems.
3	3.4	The added value of the NER was not shown in the experiment because of the lack of named entities related to concerts.
4	3.5	The ER did not identify duplicates in the experiment.

Table 5.2: Experiment limitations.

Bibliography

[ACD$^+$98] James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic detection and tracking pilot study final report. In *IN PROCEEDINGS OF THE DARPA BROADCAST NEWS TRANSCRIPTION AND UNDERSTANDING WORKSHOP*, pages 194–218, 1998.

[APL98] James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–45. ACM, 1998.

[BGMM$^+$09] Omar Benjelloun, Hector Garcia-Molina, David Menestrina, Qi Su, Steven Euijong Whang, and Jennifer Widom. Swoosh: A generic approach to entity resolution. *The VLDB Journal*, 18(1):255–276, January 2009.

[BMSV13] Paolo Boldi, Andrea Marino, Massimo Santini, and Sebastiano Vigna. Bubing: Massive crawling for the masses. 2013.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *COMPUTER NETWORKS AND ISDN SYSTEMS*, pages 107–117. Elsevier Science Publishers B. V., 1998.

[BSW99] Daniel M Bikel, Richard Schwartz, and Ralph M Weischedel. An algorithm that learns what's in a name. *Machine learning*, 34(1-3):211–231, 1999.

[CGHO] Gordon V Cormack, Maura R Grossman, Bruce Hedin, and Douglas W Oard. Overview of the trec 2010 legal track.

[CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[DS04] Prasanna Desikan and Jaideep Srivastava. Analyzing network traffic to detect e-mail spamming machines. In *ICDM workshop on privacy and security aspects of data mining*, 2004.

[Fac14] Facebook. Facebook reports first quarter 2014 results, 06 2014.

[FGM05] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *In ACL*, pages 363–370, 2005.

[Hoy09] L. H. Hoyle. Event marketing: How to successfully promote events, festivals, conventions, and expositions. In *Web 2.0 Summit*, 2009.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning. *Elements*, 1:337–387, 2009.

[KA04] Giridhar Kumaran and James Allan. Text classification and named entities for new event detection. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 297–304. ACM, 2004.

[Kle99] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *JOURNAL OF THE ACM*, 46(5):604–632, 1999.

[Kru95] Philippe B. Kruchten. The 4+1 view model of architecture. *IEEE SOFTWARE*, 12:42–50, 1995.

[LM02] Ninghui Li and John C. Mitchell. Design of a role-based trust management framework. In *In Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 114–130. IEEE Computer Society Press, 2002.

[Mar00] S. P. Marsh. Formalizing trust as a computational concept. In *Ph.D. Thesis*. University of Stirling, 2000.

[MC02] D. Harrison Mcknight and Norman L. Chervany. What trust means in e-commerce customer relationships: An interdisciplinary conceptual typology. *International Journal of Electronic Commerce*, 6:35–59, 2002.

[Nai14] Vineeth G Nair. *Getting Started with Beautiful Soup*. Packt Publishing Ltd, 2014.

[NFSC13] J. Jenkins K. Padmanabhan N. F. Samatova, W. Hendrix and A. Chakraborty. *Practical Graph Mining with R*. Chapman and Hall/CRC, 2013.

[PG07] Fernando Pérez and Brian E. Granger. IPython: a system for interactive scientific computing. *Computing in Science and Engineering*, 9(3):21–29, May 2007.

[PPK12] Georgios Petkos, Symeon Papadopoulos, and Yiannis Kompatsiaris. Social event detection using multimodal clustering and integrating supervisory signals. In *Proceedings of the 2Nd ACM International Conference on Multimedia Retrieval*, ICMR '12, pages 23:1–23:8, New York, NY, USA, 2012. ACM.

[PSM$^+$12] Symeon Papadopoulos, Emmanouil Schinas, Vasileios Mezaris, Raphaël Troncy, and Ioannis Kompatsiaris. Social event detection at MediaEval 2012: Challenges, dataset and evaluation. In *MEDIAEVAL 2012, CEUR Workshop Proceedings, Volume 927, Multimedia Benchmark Workshop Benchmarking Initiative for Multimedia Evaluation, Pisa, Italy, October 4-5, 2012*, Pisa, ITALY, 10 2012.

[RC12] Timo Reuter and Philipp Cimiano. Event-based classification of social media streams. In *Proceedings of the 2Nd ACM International Conference on Multimedia Retrieval*, ICMR '12, pages 22:1–22:8, New York, NY, USA, 2012. ACM.

[Rea96] J.M. Reagle. Trust in a cryptographic economy and digital security deposits: Protocols and policies. In *Master Thesis*. Massachusetts Institute of Technology, 1996.

[RK07] Min Ragan-Kelley. Interactive parallel computing with ipython and pyzmq. 2007.

[RPP$^+$13] Timo Reuter, Symeon Papadopoulos, Giorgos Petkos, Vasileios Mezaris, Yiannis Kompatsiaris, Philipp Cimiano, Christopher de Vries, and Shlomo Geva. Social event detection at mediaeval 2013: Challenges, datasets, and evaluation. Proceedings of the MediaEval 2013 Multimedia Benchmark Workshop Barcelona, Spain, October 18-19, 2013, 2013.

[SD06] Parag Singla and Pedro Domingos. Entity resolution with markov logic. In *In ICDM*, pages 572–582. IEEE Computer Society Press, 2006.

[SH02] Bomil Suh and Ingoo Han. Effect of trust on customer acceptance of internet banking. *Electronic Commerce Research and Applications*, 1:247–263, 2002.

[Sow91] John F. Sowa. Principles of semantic networks. Morgan Kaufmann, 1991.

[SS00] Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168, 2000.

[SWC13] S. Qasmi H. Sachwani R. Lalani S. Wasi, Z. Shaikh and A. Chagani. Event information extraction system (eiee): Fsm vs hmm. In *International Journal of Computer, Information Science and Engineering*, volume 7, 2013.

[Tea08] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[TO09] J. Battelle T. O'Reilly. Web squared: Web 2.0 five years on. In *Web 2.0 Summit*, 2009.

[TSK05] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[Uni14] International Telecommunication Union. Percentage of individuals using the internet, 04 2014.

[WL11] Jianshu Weng and Bu-Sung Lee. Event detection in twitter. In *ICWSM*, 2011.

[YY98] Jaime Carbonell Yiming Yang, Tom Pierce. A study on retrospective and online event detection. In *SIGIR'98*, 1998.

[ZC13] Xiangmin Zhou and Lei Chen. Event detection over twitter social media streams. *The VLDB Journal*, pages 1–20, 2013.

[ZM00] Giorgos Zacharia and Pattie Maes. Trust management through reputation mechanisms. In *Special Issue on Trust, Deception and Fraud in Agent Societies*, pages 881–907, 2000.

[ZZW07] Kuo Zhang, Juan Zi, and Li Gang Wu. New event detection based on indexing-tree and named entity. *Annual ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 215–222, 2007.